HOCHSCHULE
MITTWEIDA
UNIVERSITY OF
APPLIED SCIENCES

BACHELORARBEIT

Frau
Cora Mayerhofer

Analyse des Renesas Synergy S7G2 Mikrocontrollers in Bezug auf Sicherheitsmerkmale

2018

Fakultät **Angewandte Computer- und Biowissenschaften**

BACHELORARBEIT

Analyse des Renesas Synergy S7G2 Mikrocontrollers in Bezug auf Sicherheitsmerkmale

Autorin:
Cora Mayerhofer

Studiengang:
Allgemeine und Digitale Forensik

Seminargruppe:
FO15w1-B

Erstprüfer:
Prof. Dr. rer. nat. Christian Hummert

Zweitprüfer:
Dipl.-Inf. Andreas Sommer

Mittweida, 2018

Faculty of **Applied Computer Sciences and Biosciences**

BACHELOR THESIS

Analysis of the Renesas Synergy S7G2 microcontroller with regards to security features

Author:
Cora Mayerhofer

Course of Studies:
General and Digital Forensic Science

Seminar Group:
FO15w1-B

First Examiner:
Prof. Dr. rer. nat. Christian Hummert

Second Examiner:
Dipl.-Inf. Andreas Sommer

Mittweida, 2018

Bibliographische Angaben

Mayerhofer, Cora: Analyse des Renesas Synergy S7G2 Mikrocontrollers in Bezug auf Sicherheitsmerkmale. 69 Seiten, 19 Abbildungen, 2 Tabellen, Hochschule Mittweida, University of Applied Sciences, Fakultät Angewandte Computer- und Biowissenschaften.

Bachelorarbeit, 2018

Referat

Das Ziel der vorliegenden Bachelorarbeit besteht darin, den Renesas Synergy S7G2 Mikrocontroller und dessen Merkmale und Funktionen in Bezug auf Sicherheitsmechanismen zu analysieren, um eventuelle Schwächen zu identifizieren. Dafür werden zu Beginn Mikrocontroller und deren Eigenschaften grundlegend erklärt. Weiterführend werden sowohl praktische als auch theoretische Untersuchungen und Betrachtungen des S7G2s angestellt, wobei das Hauptaugenmerk in der Praxis auf der JTAG-Schnittstelle liegt. Abschließend wird der Renesas Controller und dessen Merkmale mit denen des Atmel ATXMEGA-A3BU-Xplained Mikrocontrollers verglichen, um die Sicherheit beider zu diskutieren.

Bibliographic Information

Mayerhofer, Cora: Analysis of the Renesas Synergy S7G2 microcontroller with regards to security features. 69 pages, 19 figures, 2 tables, Hochschule Mittweida, University of Applied Sciences, Faculty of Applied Computer Sciences and Biosciences.

Bachelor Thesis, 2018

Abstract

The aim of this Bachelor Thesis is to analyze the Renesas Synergy S7G2 microcontroller and its functions and features in relation to security mechanisms in order to identify possible security vulnerabilities. At the beginning, microcontrollers and their properties are explained with their basic functions. Furthermore, practical as well as theoretical examinations of the S7G2 are carried out. In practice, mainly focused on the JTAG interface. Finally, the Renesas controller and its features are compared with those of the Atmel ATXMEGA-A3BU-Xplained microcontroller to discuss the safety of both.

I Inhaltsverzeichnis

II Abbildungsverzeichnis

III Tabellenverzeichnis

IV Abkürzungsverzeichnis

μC ..... Mikrocontroller
AES ..... Advanced Encryption Standard
API ..... Application Programming Interface
BLE ..... Bluetooth Low Energy
BSDL ..... Boundary Scan Description Language
BSP ..... Board Support Package
BSR ..... Boundary-Scan-Register
CAN ..... Controller Area Network
CPLD ..... Complex Programmable Logic Device
CPU ..... Central Processing Unit
CRC ..... Cyclic Redundancy Check
CRCSWR ..... Snoop-On-Write/Read Switch
DES ..... Data Encryption Standard
DMAC ..... Direct Memory Access Controller
DNS ..... Domain Name System
DR ..... Data Register
EEPROM ..... Electrically Erasable Programmable Read Only Memory
FPGA ..... Field Programmable Gate Array
FSF ..... Free Software Foundation
FSTN ..... Film Super Twisted Nematic
FTP ..... File Transfer Protocol
GDB ..... GNU Debugger
GND ..... Ground
GNU ..... GNU's Not Unix
GPIO ..... General Purpose Input/Output

HAL ..... Hardware Abstraction Layer

HTML ..... Hypertext Markup Language

HTTP ..... Hypertext Transfer Protocol

I/O ..... engl. I/O – Input/Output

IEEE ..... Institute of Electrical and Electronic Engineers

IIC ..... Inter-Integrated Circuit

IoT ..... Internet of Things

IP ..... Internet Protocol

IR ..... Instruction Register

IRQ ..... Interrupt Request

JTAG ..... Joint Test Action Group

kB ..... Kilobyte

kHz ..... Kilohertz

LCD ..... Liquid Crystal Display

LIN ..... Local Interconnect Network

LSB ..... Least Significant Bit

LTE ..... Long Term Evolution

M2M ..... Machine-to-Machine

MAC-Adresse .. Media-Access-Control-Adresse

MB ..... Megabyte

MCU ..... Microcontroller Unit

MIPS ..... Million Instructions per second

MPU ..... Memory Protection Unit

MQTT ..... Message Queue Telemetry Transport

MSB ..... Most Significant Bit

MSP ..... Main Stack Pointer

PDI ..... Program and Debug Interface

Pmod ..... Peripheral Module Interface

POP3 ..... Post Office Protocol v3

PRC ..... Protect

PRCR ..... Protect-Register

PSP ..... Process Stack Pointer

QSPI ..... Quad serial peripheral interface

QVGA ..... Quadruple Video Graphics Array

RAM ..... Random Access Memory

RISC ..... Reduced Instruction Set Computer

RMII ..... Reduced Media Independent Interface

ROM ..... Read Only Memory

RTOS ..... Real-Time Operating System

SCI ..... Serial Communication Interface

SHA ..... Secure Hash Algorithm

SMTP ..... Simple Mail Transfer Protocol

SoC ..... System-on-Chip

SP ..... Stack Pointer

SPI ..... Serial Peripheral Interface

SRAM ..... Static Random Access Memory

SSP ..... Synergy Software Package

SWD ..... Serial Wire Debug

TAP ..... Test Access Port

TCP ..... Transmission Control Protocol

TFT ..... Thin-film Transistor

TLS ..... Transport Layer Security

TPU ..... Time Processing Unit

UDP ..... User Datagram Protocol

USART ..... Universal Synchronous/Asynchronous Receiver Transmitter

USB ..... Universal Serial Bus

VCC ..... Voltage at the common Collector

WDT ..... Watchdog Timeout

WLAN ..... Wireless Local Area Network

WWW ..... World Wide Web

1 Einleitung

Mikrocontroller gewinnen laut [BU10] in der heutigen Zeit immer mehr an Bedeutung und sind aus dem kontemporären Alltag nicht mehr wegzudenken. Sie werden häufig in der Gestalt von eingebetteten Systemen in nahezu allen denkbaren Bereichen des Lebens eingesetzt. Sei es im Bereich der Industrie – z.B. in Heizungssteuerungen oder in verschiedener Sicherheitstechnik, in der Automobilbranche – z.B. im Airbag oder dem Motor, im Bereich der Unterhaltungselektronik wie Smartphones und Fernsehgeräten oder in ganz alltäglichen Gebrauchsgegenständen, wie beispielsweise Waschmaschinen, Toastern oder Geldkarten.

1.1 Motivation

Aufgrund des immer größer werdenden Einflusses von Mikrocontrollern und den darauf gespeicherten Daten und Informationen kommt es zunehmend zu Angriffen auf diese eingebetteten Systeme. Daraus folgt, dass eine der wichtigsten Anforderung an diese Einheiten im Bereich der Sicherheit und des Schutzes liegt. Viele der genannten Geräte weisen Sicherheitslücken auf, die einen Angriff durch Dritte erleichtern.

1.2 Zielstellung

Um solche eventuellen Sicherheitslücken aufzudecken, setzt sich diese Bachelorarbeit mit den allgemeinen Grundlagen über Mikrocontroller und im Speziellen mit den Funktionsweisen und Merkmalen des Renesas Synergy S7G2s auseinander. Dieser spezielle Mikrocontroller soll weiterführend auf Sicherheitsmerkmale überprüft werden, um festzustellen, wo sich eventuelle Angriffspunkte befinden und wie man diese ausnutzen kann.
Dies ist besonders für die Forensik von sehr großem Interesse, da in diesem Fall Einblicke und besonders der Zugriff auf Daten, welche an den verschiedensten Orten gespeichert sein können, die wichtigste Arbeitsgrundlage darstellen. Durch eventuell auftretende Lücken kann man in das System eindringen und sich Zugang zu den darauf befindlichen Daten verschaffen. Besonders die Tatsache, dass es für viele Schwach-

stellen keine Updates gibt und sie so während der Nutzungsdauer nur wenig bis gar nicht behoben werden, spielt im Hinblick auf die Sicherheit eine sehr große Rolle.
Nachdem die Identifizierung vorhandener Sicherheitsmerkmale durchgeführt wurde, sollen diese in ihrer Funktionalität erklärt werden. Abschließend soll der spezielle Mikrocontroller der Firma Renesas mit dem Atmel ATXMEGA-A3BU-Xplained Mikrocontroller verglichen werden, um so dessen Vor- und Nachteile darstellen zu können.

2 Grundlagen

2.1 Mikroprozessoren, Mikrorechner und Mikrocontroller

2.1.1 Begriffserklärungen und Abgrenzung

Zunächst werden in diesem Kapitel einige grundlegende Begriffe genauer definiert.
Ein Mikroprozessor ist die Zentraleinheit (engl. CPU – Central Processing Unit) eines Datenverarbeitungssystems.
Die CPU ist heutzutage in den meisten Fällen mit weiteren Komponenten auf einem Chip untergebracht. Dieser besteht im Normalfall sowohl aus einem Steuerwerk und einem Rechenwerk, welche zusammengefasst als Prozessorkern bezeichnet werden, als auch aus einer Schnittstelle zur Außenwelt. Je nach Prozessor können weitere Komponenten wie z.B. virtuelle Speicherverwaltung vorhanden sein. Als Herzstück eines Computers dient die zentrale Recheneinheit dazu, in einem regelmäßigen Rhythmus, auch als Taktfrequenz bezeichnet, Befehle abzuarbeiten und auszuführen. Wie es die Bestandteile bereits vermuten lassen, sind die Hauptaufgaben eines Mikroprozessors Rechnen und Steuern.
Innerhalb des Rechenwerks führt die zentrale Recheneinheit verschiedenste Rechenoperationen aus, wohingegen sie im Steuerwerk die Zusammenarbeit der einzelnen Komponenten gewährleistet. So speichert und liest sie Daten aus dem Arbeitsspeicher (engl. RAM – Random Access Memory) und verarbeitet sowohl Ein- als auch Ausgaben von peripheren Geräten, wie der Maus, dem Drucker oder der Tastatur.
Ein Mikrorechner bzw. –computer ist eine Datenverarbeitungsanlage, deren Zentraleinheit aus einem oder mehreren Mikroprozessoren besteht. Zudem beinhaltet ein Mikrorechner sowohl einen Speicher sowie Ein- und Ausgabeschnittstellen als auch ein Verbindungssystem, welches Daten zwischen den einzelnen Komponenten überträgt [Czi15].

Ein Mikrorechner- oder Mikrocomputersystem ist ein Rechner bzw. Computer, bei dem die angeschlossenen Peripherie-Geräte bereits an die Ein- und Ausgabeschnittstellen und nachfolgend an die zentrale Verarbeitungseinheit angeschlossen sind [BU10].

Die Abgrenzung der bereits definierten Begriffe wird in folgender Abbildung sichtbar:

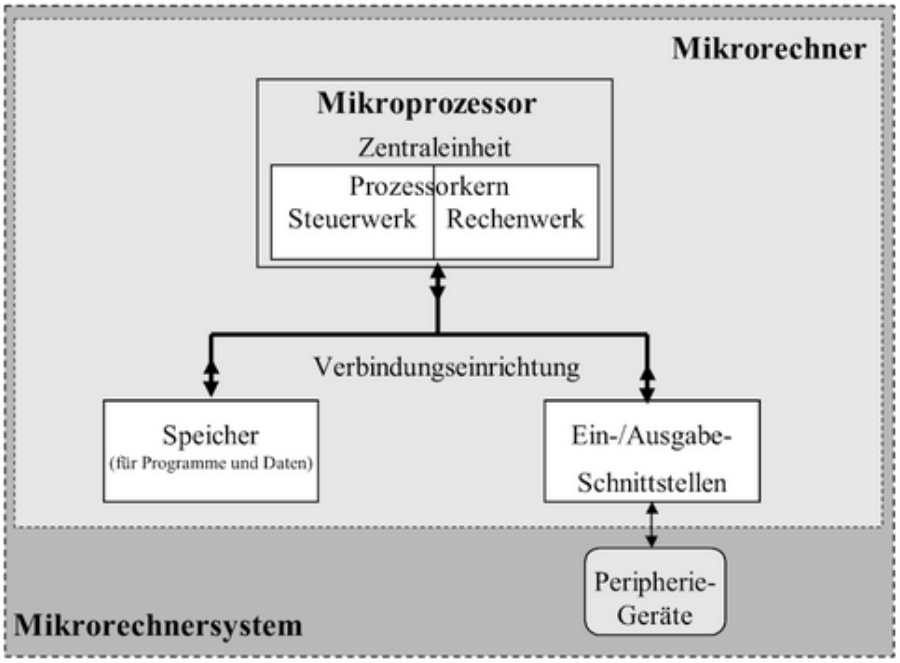

Abbildung 2.1: Abgrenzung von Mikroprozessoren, Mikrorechnern und Mikrorechnersystemen [BU10]

Ein Mikrocontroller (engl. MCU – Microcontroller Unit) vereint die Komponenten eines Mikrorechners auf nur einem Chip, um die Steuerungs- und Kommunikationsaufgaben mit möglichst wenigen Bausteinen, aber dennoch leistungsfähig, zu lösen. Deshalb werden sie auch Ein-Chip-System (engl. SoC – System-on-Chip) genannt.
MCUs sind Halbleiterchips, welche einen Prozessor und zugleich auch komplexe Peripheriemodule wie z.B. CAN[1], LIN[2], USB oder SPI[3] beinhalten.
Da heutige MCUs je nach Aufgabenstellung vielfältig eingesetzt werden können, existiert eine Vielzahl verschiedener anwendungsorientierter Mikrocontroller, welche die Anzahl der Mikroprozessoren inzwischen um ein Weites übertrifft [BU10].

[1]Controller Area Network = serielles Bussystem, welches die einzelnen Komponenten miteinander verbindet und genutzt wird, um Kabelbäume zu reduzieren und damit Gewicht zu sparen. Wird häufig innerhalb der Fahrzeugelektronik verwendet.
[2]Local Interconnect Network = serielles Kommunikationssystem, für die Vernetzung von Sensoren und Aktoren, beispielsweise in Kraftfahrzeugen.
[3]Serial Peripheral Interface = Bus-System, mit dem digitale Schaltungen nach dem Master-Slave-Prinzip miteinander verbunden werden können.

2.1.2 Eigenschaften von Mikrocontrollern

Manche der hier aufgeführten Eigenschaften eines Mikrocontrollers wurden bereits beschrieben:

- Er bearbeitet meist feste Programme für feste Anwendungen
- Er besitzt als Programmspeicher einen Festwertspeicher, welcher nach der Beendigung der Stromzufuhr seinen vorherigen Inhalt nicht verliert
- Der Platzbedarf für die Prozessorsteuerung wird reduziert, da viele Komponenten auf einem Chip integriert werden. Je nach Aufgabenstellung kann dadurch die Komponentenanzahl erhöht, oder die Platinengröße verringert werden.
- Es besteht ein geringerer Stromverbrauch als bei universellen Mikroprozessoren, da MCUs Power-Down- bzw. Sleep-Modi besitzen. Dies bedeutet, dass verschiedene Funktionsblöcke, wie z.B. der Timer, deaktiviert werden, um Strom zu sparen. Infolgedessen wird eine möglichst lange Laufzeit mit einer möglichst kleinen Batterie ermöglicht.

Diese Eigenschaften führen sowohl im laufenden Betrieb als auch bei der Baustein- und Platinenherstellung zu einer enormen Kostenreduktion. Zudem ermöglichen sie schnelle und individuelle Design- und Programmänderungen. Durch die vergleichsweise geringe Anzahl fehleranfälliger externer Komponenten wird sowohl die Wartungsfreundlichkeit des Geräts als auch die Betriebssicherheit erhöht.
Dies geschieht durch spezielle Bausteine des Controllers, wie beispielsweise Überwachungsschaltungen (engl. WDT – Watchdog Timer)[4] oder den Power-Down- bzw. Sleep-Modus. Weiterhin verhindert ein Hardware-Kopierschutz für den Chip-internen Speicher das unautorisierte Kopieren von kostenpflichtiger Software.
In welcher Branche und Form ein Mikrocontroller eingesetzt wird, ist von der Aufgabenstellung abhängig. Jedoch gibt es mehrere Haupteinsatzgebiete von Mikrocontrollern:

- Kommunikation und Datenfernübertragung, z.B. in Smartphones
- Auto und Verkehr, z.B. in Autoschlüsseln oder im Anti-Blockier-System (ABS)
- Konsumelektronik, z.B. in Spielzeug oder Fernsehgeräten
- SmartCard/Chipkarten-Anwendungen, z.B. in Geld- oder Krankenkassenkarten

[4] Watchdog-Bausteine überwachen die Funktion anderer elektronischer Komponenten, indem sie von den zu überwachenden Komponenten regelmäßig ein Signal empfangen [SGD09].

- Industrieanwendungen, z.B. in Klimaanlagen oder Heizungssteuerungen
- Computerperipherie, z.B. in Grafikkarten oder Hard-Disks [Bäh10]

2.1.3 Typischer Aufbau eines Mikrocontrollers

In der Abbildung 2.2 ist der typische Aufbau eines Mikrocontrollers grob dargestellt:

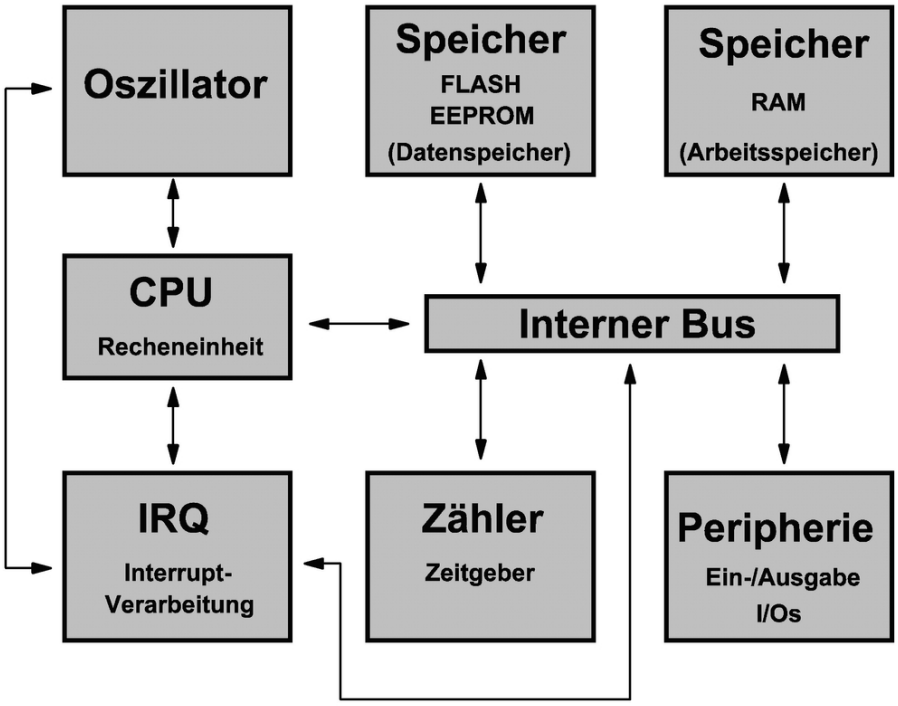

Abbildung 2.2: Prinzipieller Aufbau eines Mikrocontrollers [Som12]

Ein Mikrocontroller ist mit den folgenden Komponenten bzw. Gruppen von Komponenten ausgestattet:

CPU Diese ist, wie bereits thematisiert, als zentrale Recheneinheit die wichtigste Komponente. Sie bearbeitet Befehle und arithmetische Operationen und wertet Signale aus.

Arbeits- und Programmspeicher Die Ablage von Programmen und Daten kann entweder in einem gemeinsamen oder einem getrennten Speicher erfolgen, wobei in der Regel der zweite Fall eintrifft. Dabei wird das Benutzerprogramm im nicht flüchtigen Flash-Speicher, dem Programmspeicher (engl. ROM – Read Only Memory), abgelegt. Im Arbeitsspeicher werden temporär Rechen-, Mess- und Steuerwerte gespeichert. Abgelegt werden beispielsweise Daten über die

berechneten Programmlaufzeiten. Normalerweise ist der RAM kleiner, aber dafür deutlich schneller als der Flash-Speicher. Zudem sind die abgelegten Werte und Daten im Gegensatz zum Flash-Speicher flüchtig, d.h., dass die Daten im RAM nach einem Neustart des Controllers nicht mehr vorhanden sind.

Peripherie Das Wort Peripherie beinhaltet mögliche Komponenten eines Mikrocontrollers, welche nicht durch Speicherbausteine oder die zentrale Recheneinheit abgedeckt sind. Dazu gehören beispielsweise digitale oder analoge Ein- und Ausgänge (kurz I/O für engl. Input/Output) oder jegliche andere Komponenten, die eine Schnittstelle zur Außenwelt darstellen.

Interner Bus Wie in der Abbildung 2.2 sichtbar, wird der Systembus für den internen Datentransport zwischen den einzelnen Funktionseinheiten des Mikrocontrollers benötigt.

Oszillator Der Oszillator wird genutzt, um elektrische Schwingungen zu erzeugen. Verluste des durch die Schaltung erzeugten Signals, können vom Oszillator selbstständig ausgeglichen werden. Das Ergebnis ist eine konstante Schwingung. Je nach Oszillatortyp kann das erzeugte Signal unterschiedliche Formen, wie z.B. Rechteck, Sinus oder Sägezahn aufweisen.

IRQ Bei einer Unterbrechungsanforderung (engl. IRQ – Interrupt Request) wird der CPU mitgeteilt, wenn es zu einer Änderung im Rechenvorgang kommt. Dies geschieht, wenn eine Komponente die Rechenleistung der Zentraleinheit benötigt und diese über einen IRQ anfordert [ITW17].

Timer Wie das Wort es bereits sagt, haben Timer die Aufgabe, eine Ganzzahl je nach Betriebsmodus zu dekrementieren oder zu inkrementieren, d.h. entweder auf- oder abwärts zu zählen. Dadurch kann man z.B. das Zählen und Erfassen von bestimmten Ereignissen realisieren [rnw14]. Zudem ist eine weitere wichtige Funktion die Überwachung des Prozessorzustands als Watchdog Timer, sowie die Verwendung als Echtzeituhr (engl. TPU – Time Processing Unit) [Bäh10].

2.2 Embedded Systems

Ein sehr großer Vorteil des Mikrocontrollers ist seine geringe Größe. Aufgrund dessen kann er direkt in das zu steuernde System integriert werden (siehe Abbildung 2.3) und mit diesem in Wechselwirkung stehen.
Deshalb nennt man solche Controller auch „eingebettete Controller“. Werden diese wiederum mit weiteren Funktionsbausteinen, wie beispielsweise Speicher- oder Ein- und Ausgabemodulen zu einem Mikrorechnersystem ergänzt, spricht man dann auch von „eingebetteten Systemen“ (engl. Embedded Systems). Die von ihnen gesteuerten Geräte nennt man „eingebettete Anwendungen“ (engl. Embedded (System) Applications) [BU10]. Hierbei übernimmt der Rechner meist Überwachungs-, Steuerungs- oder Regelungsfunktionen oder ist oft auch für eine Form der Daten- bzw. Signalverarbeitung zuständig [SG17].
In der Regel sind in komplexen eingebetteten Systemen mehrere Komponenten an Mikrocontrollern oder digitalen Signalprozessoren (engl. DSP – Digital signal pocessor), wie in der Abbildung 2.3 ersichtlich wird, verbaut.

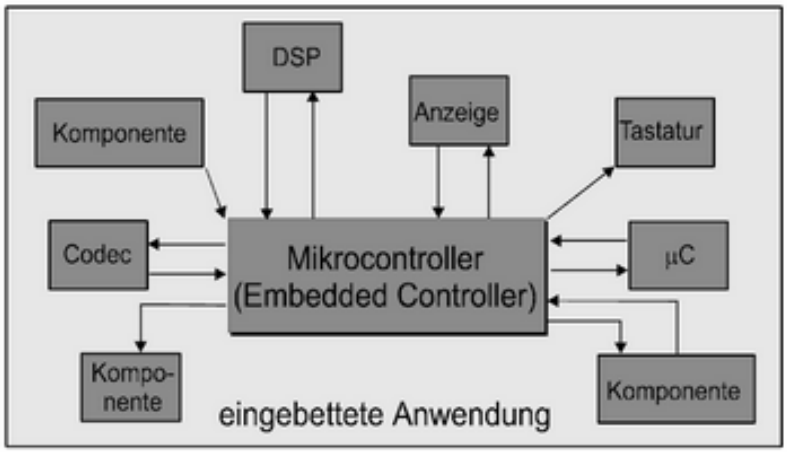

Abbildung 2.3: Einbettung eines Mikrocontrollers in ein System [BU10]

Herausforderungen im heutigen Embedded System Design

Eingebettete Systeme haben sich in den letzten Jahrzehnten enorm verändert. Vor der Jahrhundertwende verwendeten diese Einheiten einige einfache Schnittstellen wie Eingabeschaltflächen oder ein Zeichen-LCD zur Ausgabe und stellten damit eine einzige Funktion bereit, die meist als einfache Schleife implementiert war. Früher konnte demnach nur eine begrenzte Anzahl an Aufgaben bewältigt werden. Ein Mikrocontroller mit einigen wenigen MIPS (Millionen Instruktionen pro Sekunde) und ein paar

kB Speicher wäre für diese Art von System ausreichend gewesen.

Die heutigen eingebetteten Systeme sind jedoch stark miteinander verbunden und weisen eine Vielzahl von Schnittstellen wie Ethernet, drahtlose oder grafische Benutzerschnittstellen auf, die alle konfiguriert werden müssen und die die Kommunikation untereinander und mit der Außenwelt gewährleisten müssen. Dies kann eine MCU mit einer Taktfrequenz von 100 MHz oder mehr, mehreren MB Flash-Speicher und einem großen RAM erfordern [Oed17].
Wie bereits an den Einsatzgebieten von Mikrocontrollern ersichtlich wird, sind die Menschen der heutigen Zeit zunehmend im alltäglichen Leben von Embedded Systems umgeben, sodass sehr viele Anforderungen an die Systeme gestellt werden. Neben der Schnittstellenanforderung, dass es möglichst eine große Auswahl an verschiedenen Ein- und Ausgabeschnittstellen geben soll, ergibt sich beispielsweise auch die Zuverlässigkeitsanforderung. Da man sich heutzutage oft auf elektronische Steuerungen und Geräte verlässt, müssen sie allzeit zuverlässig funktionieren — z.B. die Bremsanlage eines Kraftfahrzeugs. Eine weitere sehr wichtige Anforderung, kontemporär möglichlicherweise eine der wichtigsten, die an eingebettete Systeme gestellt wird, ist die zeitliche Anforderung. Bestimmte Tätigkeiten müssen innerhalb des Systems in einer vorgeschriebenen Zeit geschehen. Betriebssysteme, die diese Bedingung erfüllen, nennt man auch Echtzeitsystem (engl. RTOS – Real-Time Operating System) [BU10].

Ein Echtzeitbetriebssystem kann hilfreich sein, wenn mehrere Threads[5] gleichzeitig priorisiert und ausgeführt werden müssen [Oed17].
Weiterführend müssen eingebettete Systeme die Anforderung der Verfügbarkeit erfüllen, d.h., dass das System jederzeit korrekt arbeiten muss. Damit zusammenhängend wird auch deutlich, dass die Wartbarkeit eine weitere wichtige Anforderung darstellt, da ein eventuell auftretender Fehler bzw. das komplett ausgefallene System innerhalb einer möglichst kurzen Zeitspanne repariert werden kann und, dass ein ausgefallenes System bestenfalls keinen Schaden verursacht.
Abschließend ist die Gewährleistung der Sicherheit und der Integrität eine weitere ausschlaggebende Anforderung, da Hackerangriffe und Datendiebstahl immer häufiger vorkommen. Aufgrund dessen wird sehr viel Wert auf die Sicherheit der Embedded Systems gelegt. Es ist wichtig, dass vertrauliche Daten geschützt werden und die Authentizität der Kommunikation gewährleistet wird [Mar08].

[5]Thread = Teil eines Programms, welcher bestimmte Anweisungen enthält

2.3 Der Renesas Synergy S7G2 Mikrocontroller

Eine Firma, die sich mit diesen Herausforderungen beschäftigt, ist die japanische Firma Renesas Electronics Corporation, welche viele verschiedene Hard- und Software anbietet [Ren18b].
Im Folgenden soll der Renesas Synergy S7G2 Mikrocontroller mit seinen Komponenten und Funktionen näher beschrieben werden.

2.3.1 Merkmale und Funktionen

Der Renesas Synergy S7G2 Mikrocontroller umfasst folgende Merkmale und Funktionen:

- **Display:**
  - 2,4 Zoll TFT-LCD-QVGA-Farbdisplay mit Touchscreen (320x240), das den internen SRAM als Display-Framebuffer-Speicher verwendet
- **Erweiterung:**
  - mit Arduino UNO Shield kompatiblem Stecker
  - 2 Pmod-Anschlüsse
- **Drahtgebundene Konnektivität:**
  - USB- und Micro-USB-Anschlüsse
  - Ethernet mit Unterstützung von RMII und IEEE 1588 Precision Time Protocol (PTP)
  - RS232/RS485 verfügbar auf Pins mit Transceiver
  - CAN verfügbar auf Pins mit Transceiver
- **Drahtlose Konnektivität:**
  - BLE (Bluetooth Low Energy) 4.0 on-board

- **Speicher:**
  - 4 MB Code Flash
  - 64 kB Data Flash
  - 640 kB SRAM
  - 8 MB QSPI Flash
- **Sicherheit:**
  - Verschlüsselung mit Algorithmen wie AES oder RSA
  - Hash-Validierung mit Verfahren wie SHA
  - Hardwarediagnose, Speicherschutz und Degugging mit Hilfe des Watchdog Timers und Programmierung mit J-Link On-Board

Im Folgenden werden verschiedene Bestandteile und deren Funktionen näher erklärt. Dabei werden nur jene berücksichtigt, die sicherheitsrelevant sind bzw. sein könnten – beispielsweise der Speicher, aufgrund der dort abgelegten Daten, oder die Peripheriegeräte und Schnittstellen, angesichts der unterschiedlichen Zugriffsmöglichkeiten.

CPU Der S7G2 Mikrocontroller besitzt einen ARM Cortex-M4 Mikroprozessor mit einer Taktrate von 240 Mhz. Zudem gibt es zur Behandlung von möglichen Ausnahmefällen und Unterbrechungen zwei kapazitive Touch-Tasten, die an Pins angeschlossen sind, welche Interrupts erzeugen können [Ren15].

Speicher Auf dem Chip des S7G2s befindet sich ein ROM, welcher einen nicht flüchtigen Festwertspeicher darstellt und vom Prozessor lediglich gelesen, jedoch nicht beschrieben werden kann [Bäh10]. Neben 4 MB Code Flash und 64 kB Data Flash verfügt der S7G2 über einen 8 MB großen QSPI Speicher.
Hierbei stellt QSPI eine Schnittstelle zur Kommunikation mit externen Flash-Speichern durch den SPI Bus dar [Nor18]. Zudem hat der S7G2 Mikrocontroller einen 640 kB großen SRAM, welcher zu den Bausteinen mit einem flüchtigen Speicher gehört [Ren15].

Energieversorgung Die Stromversorgung des S7G2s erfolgt über den Debug-USB-Anschluss. Sobald der Controller mit Spannung versorgt wird, setzt der Power-On-Reset-Monitor den Mikrocontroller zurück. Der Speicher, von dem der S7G2 anschließend startet, hängt vom Jumper J1 ab.
Wenn dieser standardmäßig die Position 1-2 annimmt, dann startet der Con-

troller die Ausführung vom internen Flash-Speicher (ROM). Befindet sich der Jumper auf Position 2-3, dann wird die Ausführung im USB-Programm-Modus gestartet, so dass das Programm direkt über die USB-Schnittstelle geladen werden kann. Zudem ist es möglich, den Controller über den Jumper J2 zurückzusetzen, selbst, wenn er mit Strom versorgt ist [Ren15].

Peripheriegeräte Der S7G2 beinhaltet neben einer Ethernet-Schnittstelle ebenso zwei Standard-Pmod[6]-kompatible Schnittstellen, welche an den Anschlüssen J12 und J14 verfügbar sind. PMODA stellt dem Mikrocontroller eine SPI-Schnittstelle, drei GPIO-Leitungen (engl. GPIO – General Purpose Input/Output) und eine Interrupt-Leitung zur Verfügung. GPIO ist eine programmierbare Allzweckein– und Ausgabe, deren Verhalten, je nachdem wie sie programmiert ist, frei bestimmbar ist [Sch18].

Zusätzlich stellt PMODB, welches am J14-Anschluss verfügbar ist, einen UART, drei GPIO-Leitungen und eine Interrupt-Leitung für den S7G2 Mikrocontroller bereit. Ein Universal Asynchronous Receiver Transmitter (kurz UART) dient zum asynchronen Senden und Empfangen von Daten, d.h., dass die Übertragung, im Gegensatz zur synchronen Datenübertragung, zu einer beliebigen Zeit ausgeführt werden kann [NM05].

Weiterführend besitzt der S7G2 sowohl eine CAN-Schnittstelle, welche sich am J7-Anschluss befindet, als auch ein Inter-Integrated Circuit (kurz IIC oder I^2C). Sowohl CAN als auch I^2C sind serielle Busse, welche für die Datenübertragung genutzt werden [PM06]. Während sich am Jumper J5 ein Micro-USB Geräteanschluss befindet, welcher nicht für die Stromversorgung verwendet werden kann, ist der J6 mit einem USB-Hochgeschwindigkeits-Host-Port (480 Mbit/s) ausgestattet, welcher Strom an angeschlossene Geräte liefern kann und mit dessen Hilfe Daten von einem anderen Gerät gelesen oder an ein anderes Gerät gesendet werden können.

Der S7G2 enthält außerdem eine Arduino Shield kompatible Schnittstelle, so dass dessen Funktionalität durch Arduino Shield Boards erweitert werden kann. Außerdem enthält der S7G2 am J16 einen verstärkten Mono-Audio-Ausgang an einer standardmäßigen 3,5mm-Audiobuchse [Ren15].

[6]Pmod = Peripheral Module Interface – ist ein Standard für Peripheriegeräte, welche mit FPGAs oder Mikrocontrollern verwendet werden und existiert als Doppel- oder Vierfach-Pmod. Diese ist beim S7G2 als Doppel-Pmod aufgeteilt in PmodA und PmodB.

Schutz und Sicherheit Der S7G2 verfügt über einen J-Link On-Board-Debugger, welcher über den USB-Anschluss J19 zugänglich ist. Dieser ermöglicht das Debugging des Mikrocontrollers, ohne dass ein zusätzlicher externer Debugger erforderlich ist [SEG18a].

Zudem gibt es eine Speicherschutzeinheit (engl. MPU – Memory Protection Unit), welche in MCUs verwendet wird, um den Zugriff auf Speicherbereiche einzuschränken und dadurch die dortigen Daten zu schützen.

Generell sollten die verwendeten Speicherschutzmechanismen die Möglichkeit bieten, Adressierungsfehler zu erkennen, ohne dass die Funktion des Systems beeinträchtigt wird. Dies verhindert zusätzlich die Entstehung von Folgeschäden [Eic90].

Die Abbildung 2.4 zeigt die Position aller relevanten Board-Komponenten, die im vorherigen Abschnitt genannt und näher erläutert wurden. Weiterhin verfügt der S7G2 über mehrere Konfigurationsoptionen, die über die Jumper eingestellt werden können.

Zudem stellt er für den seriellen Datenaustausch zwischen Mikroprozessor und Peripheriegeräten den Serial Communication Interface (kurz SCI)-Port 3 der S7G2-MCU über drei verschiedene elektrische Schnittstellen zur Verfügung: RS-232, RS-485 und die MCU-Breakout-Header. Tabelle B.1 (siehe Anhang) listet die verschiedenen Jumper und ihre Positionen und Funktionen auf, während die Tabelle B.2, welche sich ebenfalls im Anhang befindet, die Optionen des RS-232-Transceivers zusammenfasst [Ren15].

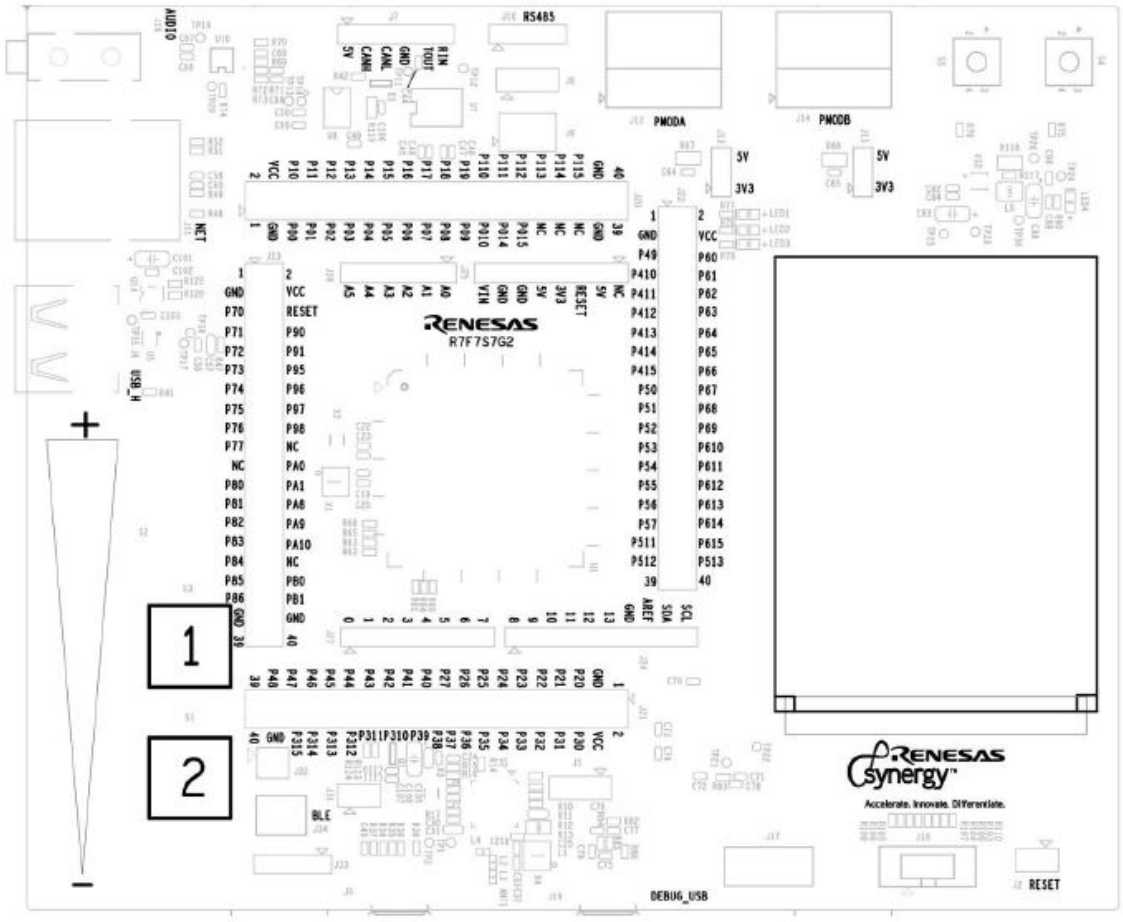

Abbildung 2.4: Layout des S7G2-Boards [Ren15]

2.3.2 Ergänzung durch Software

Um die gegebene Hardware zu ergänzen, stellt die Firma Renesas eine große Anzahl verschiedener Software zur Verfügung, welche nach Anlegen eines Online-Benutzerkontos in der Renesas Synergy Platform frei zugänglich ist.
Im Folgenden wird das Synergy Software Package (SSP) thematisiert, wobei das Hauptaugenmerk auf Funktionen liegt, die mit dem Sicherheitsaspekt verbunden sind. Zunächst wird in der Abbildung 2.5 dargestellt, welche Bestandteile im Software Package enthalten sind.

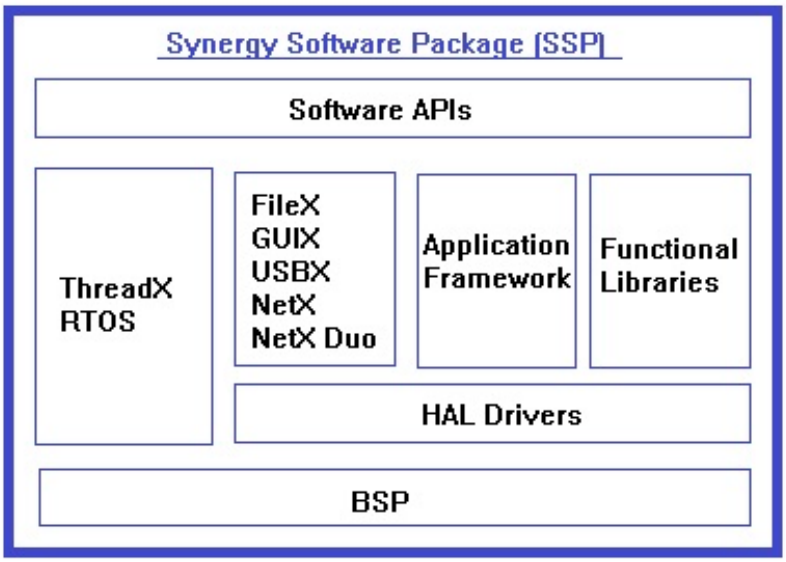

Abbildung 2.5: Bestandteile des Synergy Software Packages [eigene Arbeit, modifiziert nach [Oed17]]

RTOS-unabhängige HAL-Treiber Es werden effiziente Treiber für die Hardwareabstraktionsschicht (engl. HAL – Hardware Abstraction Layer)[7] für alle Peripheriegeräte und Systemdienste angeboten.

Application Frameworks Die Application Frameworks enthalten verschiedene Dienste auf Systemebene, die das RTOS mit der HAL verbinden, um Interprozess-Messaging, Sicherheitsdienste, serielle Kommunikation, Audiowiedergaben, kapazitive Berührungserkennung und Weiteres zu gewährleisten. Dabei reduziert die Vollständigkeit dieser Frameworks sowohl Fehler als auch Zeit bei der Entwicklung einer Anwendung.

Board Support Package (BSP) Das BSP ist dafür verantwortlich, dass die MCU auf die Hauptanwendung (die *main()*-Funktion) zurückgesetzt wird und stellt zudem Informationen zu den SSP-Modulen bereit, sodass diese automatisch für die jeweilige Hardware konfiguriert werden können.

Funktionale Bibliotheken Die funktionalen Bibliotheken enthalten zum Beispiel spezialisierte Software für digitale Signalverarbeitung oder sicherheitsrelevante und verschlüsselungsbezogene Funktionen, um die Entwicklungszeit zu reduzieren und die Stabilität der Endanwendung zu verbessern.

[7] Hardware Abstraction Layer = eine Schicht des Betriebssystems, die sowohl den Kernel als auch die Software von der Harware trennt und durch die das Betriebssystem nahezu alle Zugriffe auf die Hardware kontrolliert [Ay18].

Echtzeitbetriebssystem ThreadX Das hier verwendete RTOS, welches von Express Logic[8] entwickelt wurde, stellt einen Multitasking-Echtzeitkernel mit vorbeugender Planung und geringem Speicherbedarf dar.

X-Ware Die X-Ware umfasst Dateisysteme (FileX), grafische Benutzerschnittstellen (GUIX) sowie USB- und TCP / IP-Kommunikation (USBX, NetX, NetX Duo und NetX Secure).

2.3.2.1 NetX Duo und NetX Secure

Um die Sicherheit innerhalb der Produkte zu gewährleisten und zu verbessern, enthält die aktuelle Version 1.3.0 des Synergy Software Packages, welche 2017 veröffentlicht wurde, verschiedene Erweiterungen. Beispielsweise integriert das SSP innerhalb der X-Ware für NetX Duo zur Sicherung der Client-Server-Kommunikation über das Internet das Transport Layer Security-Protokoll (kurz TLS) und das Message Queue Telemetry Transport-Protokoll (kurz MQTT) zum Nachrichtenaustausch zwischen Geräten. Das SSP in der Version 1.3.0 bietet außerdem WLAN-, LTE-Mobilfunk- und Bluetooth-Low-Energy-Framework-Anwendungen.
Das aktuelle SSP beinhaltet außerdem NetX Secure, um die Kommunikation verbundener IoT-Geräte über öffentliche Netzwerke und das Internet zu schützen. Entwickler können NetX Secure einsetzen, um die Identitäten von Absender und Empfänger zu verifizieren und das Abhören und Manipulieren von Daten, die über das Netzwerk gesendet werden, zu verhindern. NetX Secure verwendet das TLS-Protokoll, um bei Verwendung des NetX Duo TCP / IP-Netzwerkstacks die Socket-Layer-Sicherheit[9] zu gewährleisten. TLS bietet hierbei Sicherheit auf drei Wegen: einerseits durch Einrichten von geheimen Schlüsseln zwischen dem Client und dem Server, andererseits durch Anwenden von Hash-Algorithmen zum Erkennen von Änderungen oder Fälschungen in Bezug auf Paketinhalte, und zu guter Letzt durch Verwendung von digitalen Zertifikaten.
Das MQTT-Protokoll ermöglicht eine leichte Machine-to-Machine(M2M)-Kommunikation von kleinen IoT-Geräten mit MCU-Unterstützung. Diese Kombination von TLS- und MQTT-Protokollen gewährleistet eine sichere und effiziente Kommunikation und Verbindung der Geräte mit dem Internet [Str17].

[8] Express Logic = ein in San Diego (Kalifornien) ansässiger Entwickler von Echtzeitbetriebssystemen und Middleware-Produkten für eingebettete Internet of Things-Anwendungen [Exp18].

[9] Das Secure Socket Layer-Protokoll dient der Authentifizierung und Verschlüsselung von Internetverbindungen. Es gewährleistet die Vertraulichkeit, Integritat und Authentizitat der Daten, welche zwischen Client und Server ausgetauscht werden [FRR13].

2.3.2.2 ThreadX

ThreadX, welches von Express Logic entwickelt wurde, ist das Echtzeitbetriebssystem (RTOS) der Synergy MCU-Familie. Es benötigt einen kleinen Flash-Speicher (weniger als 2 kB) und hat eine kleine RAM-Anforderung (<1 kB für Kernel-RAM).

2.3.2.3 Vorteile eines RTOS

Da moderne Embedded Systems, wie bereits thematisiert, immer höheren Ansprüchen ausgesetzt sind und vor allem reaktionsfähig sein müssen, nutzt man oftmals Echtzeitbetriebssysteme.
Bei den bekannten Standard-Betriebssystemen wie Linux oder Windows steht besonders die Korrektheit der Datenverarbeitung und der damit verbundenen Ergebnisse im Vordergrund. Bei Echtzeitsystemen kommt es aber zusätzlich darauf an, dass Zeitbedingungen erfüllt werden, d.h., dass Verarbeitungsprogramme ständig betriebsbereit sein müssen und dass deren Ergebnisse ebenfalls innerhalb einer vorgegebenen, möglichst kurzen Zeitspanne verfügbar sind. Dies ist bei Standard-Betriebssystemen in der Art nicht gegeben. Ihre Timing-Unsicherheit ist zu groß und sie sind deshalb den Echtzeitanforderungen nicht gewachsen [WB05].
Außerdem benötigen Nicht-Echtzeitsysteme durch den großen Funktionsumfang mehr Ressourcen und Speicherplatz als ein RTOS. Ein weiterer Grund dafür, dass Standard-Betriebssysteme in Embedded Systems nicht verwendet werden, ist, dass das RTOS spontan auf zufällig auftretende Ereignisse innerhalb einer definierten Zeit reagieren muss.
Das Betriebssystem, welches für Embedded Systems notwendig ist, muss sowohl schnell und klein als auch nachvollziehbar und konfigurierbar sein [Oed17].

2.3.2.4 Die Hauptmerkmale von ThreadX

Das RTOS von ExpressLogic ist, wie für Embedded Systems üblich, klein und mit einem sehr geringen RAM-Speicherbedarf ausgestattet. Zudem ist es für eine schnelle Leistung und einen geringen Overhead optimiert. Es bietet verschiedene Echtzeitplanungsalgorithmen sowie effiziente Multitasking-Routinen wie Round-Robin-Scheduling und Time Slicing oder Preemtive-Threshold-Scheduling [Oed17].
Um die Reihenfolge der zu bearbeitenden Rechenprozesse festzulegen, existieren verschiedene Strategien, welche man Scheduling nennt.
Innerhalb des sogenannten Round-Robin-Scheduling werden alle Prozesse in einer Warteschlange eingereiht und anschließend abgearbeitet, indem ihnen nacheinander

für eine bestimmte Zeitspanne der Zugang zu bestimmten Ressourcen gewährt wird. Dieser Zeitraum, für den ein Prozess in einem präemptiven Multitasking-System laufen darf, wird als Time Slicing bezeichnet [FS12]. Der Begriff präemtiv stammt aus dem Englischen und bedeutet so viel wie „jemandem zuvorkommen, um ihn von etwas abzuhalten“. Beim Nicht-präemptiven Scheduling wird ein Prozess von der CPU beachtet bis er seine Aufgaben erfüllt hat. Alle weiteren Prozesse ketten sich unterdessen in eine Runqueue (Liste der lauffähigen Prozesse) ein und werden anschließend nacheinander bearbeitet. Blockiert ein Prozess während seiner Laufzeit, beispielsweise weil notwendige Daten nicht zur Verfügung stehen, wird er von der CPU verdrängt und im Anschluss wieder an das Ende der Runqueue angekettet. Dies bedeutet, dass beim Nicht-präemptiven Scheduling ein Prozess lediglich von der CPU verdrängt wird, wenn er die CPU blockiert oder diese freiwillig aufgibt, wohingegen beim präemptiven Scheduling der Prozess auch aus anderen Gründen von der CPU unterdrückt werden kann – beispielsweise durch Überschreitung einer Zeitgrenze oder weil er von einem anderen wichtigen Prozess niedergehalten wird [Fel11].

Ressourcen innerhalb eines eingebetteten Systems sind hauptsächlich durch Zeit- und Speicheranforderungen begrenzt. ThreadX bietet mehrere Optionen, um diese zu verwalten.
So sind beispielsweise Anwendungstimer Ressourcen, die vom RTOS verwaltet werden und die eine unbegrenzte Anzahl von Software-Timern ermöglichen. Diese Timer sind in drei Betriebsmodi verfügbar: Einmalig, periodisch und relativ. Ein One-Shot-Timer ruft eine Benutzerfunktion nur einmal nach Ablauf des Timers auf, während ein periodischer Timer nach einem festgelegten Intervall wiederholt eine Benutzerfunktion aufruft. Der relative Timer ist ein einzelner kontinuierlich inkrementierender 32-Bit-Tick-Zähler. Alle Timerabläufe werden in Ticks angegeben, welche konfigurierbar sind – z. B. kann 1 Tick 10 ms entsprechen.

Die Synchronisierung von Threads und die Kommunikation zwischen Tasks ist ein weiteres großes Thema in eingebetteten Systemen, so auch innerhalb des Renesas Synergy S7G2s.
Da innerhalb eines RTOS eine Echtzeitdatenverarbeitung stattfindet, müssen die einzelnen Tasks häufig durch gemeinsame Daten oder Nachrichten untereinander Informationen austauschen. Dies erfordert Synchronisation bzw. konsistentes Verhalten und ist mit Problemen verbunden. Um diese zu lösen, nutzt ThreadX Semaphoren[10] und gegenseitige Ausschlüsse [Oed17].
Im Folgenden wird der Zusammenhang zwischen den Anforderungen, den Problemen

[10]Ein Semaphor ist eine Datenstruktur, welche zur Verwaltung beschränkter Ressourcen, auf die mehrere Prozesse oder Threads zugreifen sollen, verwendet wird.

und den Lösungsansätzen dargestellt. Nehmen wir an, dass Task 1 auf einen Speicherbereich zugreift, um Bilddaten abzuspeichern, während Task 2 diese auswertet. Wenn nun beide Tasks gleichzeitig auf den Speicher zugreifen, würde das Auswertungsprogramm eventuell auf Daten aus verschiedenen Aufnahmesituationen reagieren und das würde zu Fehlern führen.

Aufgrund dessen besteht das Ziel darin, zu verhindern, dass beide Tasks gleichzeitig auf den Bildspeicher zugreifen. Als Lösung könnte man beispielsweise den gegenseitigen Ausschluss nennen: erst Beschreiben des Bildspeichers, dann Auswerten des Bildspeichers. Daraus ergibt sich auch die Definition des kritischen Bereichs, welcher ein Codestück eines Tasks darstellt, in dem auf gemeinsam genutzte Mittel zugegriffen wird und für den aus diesem Grund Synchronisationsbedarf besteht. Beim gegenseitigen Ausschluss entsteht das Problem, dass in mehreren Tasks solche kritischen Bereiche bestehen, welche das selbe Mittel betreffen – wie im obigen Beispiel der Bildspeicher. Deshalb muss, wie bereits thematisiert, sichergestellt werden, dass mehrere Tasks nie gleichzeitig in die Abarbeitung kritischer Bereiche eintreten, damit innerhalb der Programmausführung keine Fehler entstehen.

Um dieses Problem zu lösen, besteht die Möglichkeit, ein Sperrbit, welches auch Schlossvariable genannt wird, einzusetzen, um einen kritischen Bereich vor unzulässigem Betreten zu schützen. Das Sperrbit kann hierbei entweder den Status „belegt“ oder „frei“ annehmen. Solange es den Wert „belegt“ hat, kann demnach kein anderer Task in den kritischen Bereich eintreten. Dies wird solange geprüft und durchgeführt, bis der Wert des Sperrbits „frei“ ist. Erst dann kann ein anderer Task in den kritischen Bereich eintreten. Allerdings ist auch dieses Vorgehen fehleranfällig, da es dennoch, z.B. durch ungünstige Unterbrechungen oder Task-Wechsel zum „Doppel-Aufenthalt“ zweier Tasks kommen kann. Zudem kostet das aktive Warten auf das Eintreten in einen kritischen Bereich Rechenzeit, da die Sperrbits kontinuierlich überprüft werden müssen.
Um die genannten Nachteile dieses Verfahrens zu vermeiden, werden Semaphore genutzt. Semaphore sind ganzzahlige Variablen für den Entwurf korrekter Synchronisationsprotokolle, so dass kein aktives Warten mehr erforderlich ist. Ausschlaggebend ist hierbei der aktuelle Wert der Semaphorvariablen:

- Positiver Wert x: Tasks der Anzahl x dürfen zeitgleich in den, dem Semaphor zugeordneten kritischen Bereich eintreten.
- Negativer Wert x: Es existiert eine Warteschlange der Länge x.
- Wert 0: Der kritische Bereich darf nicht betreten werden und es existiert auch keine Warteschlange [BH09].

3 Methoden

3.1 Portscan

Mit der weiteren Verbreitung verschiedenster IoT-Geräte und deren Verbindung zum Internet werden diese zu potenziellen Angriffszielen. Eine Möglichkeit zur Analyse des Ist-Zustands eines Systems stellen sogenannte Portscanner, wie beispielsweise *nmap* dar.
Mit Hilfe der IP-Adresse können Computer innerhalb des weltweiten Datennetzes identifiziert werden. Hierbei stellen Ports eine weitere logische Untergruppe dar. Jeder Dienst im Internet wird über einen bestimmten Port über UDP/IP oder TCP/IP durchgeführt. Vorhanden sind genau 65.536 Ports, welche sowohl in privilegierte als auch in unprivilegierte Ports eingeteilt sind. Dabei sind die Ports von 1 bis 1023 privilegiert, während alle anderen unprivilegierte Ports darstellen. Diese Unterscheidung stammt aus der Unix-Welt, da man für alle Ports bis 1024 Root-Rechte, also den administrativen Zugriff, benötigt.
Einige Ports, wie Port 110 (POP3), 80 (HTTP), 53 (DNS), 25 (SMTP), 21 und 20 (FTP), werden täglich bei der Nutzung des Internets verwendet. Über Port 80 TCP findet die Übertragung von Internetseiten statt, d.h. die HTTP-Verbindung. Die Ports 20 und 21 ermöglichen die Übertragung von Dateien. Während Port 53 UDP und TCP nutzt, um die Zuordnung von Namen zu IP-Adressen durchzuführen, verwendet der Port 110 TCP für die Abholung von E-Mails und der Port 25 TCP für den Versand von E-Mails. Generell gilt, dass lediglich die benötigten Ports offen sein sollten, da mehr offene Ports auch mehr Angriffspunkte bedeuten.

Anhand eines Portscans wird ersichtlich, welche Ports innerhalb eines Systems offen oder geschlossen sind und welche angebotenen Dienste dadurch verfügbar sind. Oft wird ein Portscan einem Angriff vorausgesetzt, um zu Beginn herauszufinden, welche Angriffsmöglichkeiten beim Zielsystem vorliegen. Um dies zu untersuchen, versucht der Angreifer eine Verbindung mit der anzugreifenden Einheit aufzubauen, indem er an die verschiedenen Ports des Zielsystems spezielle Datenpakete schickt und währenddessen die Antworten dessen analysiert.
Diese Datenpakete, welche an die anzugreifende Einheit geschickt werden, nennt man SYN-Pakete, da dadurch die Anforderung einer Verbindung gestellt wird. Nachdem

das Zielsystem diese Synchronisationspakete bekommen hat, bestehen drei Möglichkeiten, um auf diese zu antworten: Ist der betreffende Port geöffnet, so sendet das Zielsystem ein Acknowledgement-Paket. Dieses ACK-Paket bestätigt die angeforderte Verbindung. Eine weitere Möglichkeit ist das Senden eines RST-Pakets. Dieses Reset bedeutet, dass der betreffende Port geschlossen ist und die Verbindung vom anzugreifenden System nicht angenommen wird. Die dritte Möglichkeit wäre, dass das Zielsystem gar nicht antwortet. Ist dies der Fall, so gilt der angesprochene Port weder als offen noch als geschlossen, sondern als versteckt.
Auf diese Art und Weise können Programme, welche Portscans durchführen, in kurzer Zeit viele Informationen liefern. Dabei erhält man nicht nur Erkenntnisse darüber, welche Ports offen oder geschlossen sind, sondern beispielsweise auch darüber, wie lange ein System schon eingeschaltet ist und welches Betriebssystem auf dem System befindlich ist. Der Funktionsumfang ist je nach Programm unterschiedlich.

Da der Portscan in erster Linie zur Überprüfung des Zustandes eines Netzwerkes oder eines Zielsystems dient, könnte man meinen, dass dieses Überprüfungsverfahren nicht als strafbar gilt. Allerdings ist der Portscan rechtlich umstritten. Er kann als erster Versuch des Eindringens in ein fremdes System oder Netzwerk gewertet werden, da empfindliche Computer in ihrer Funktionalität durch viele Verbindungsanfragen gestört werden können.
Dadurch besteht die Möglichkeit, ein Zielsystem durch einen Portscan zu blockieren, so dass dieses keine Verbindungen mehr aufbauen kann. In Deutschland gilt dies als Angriff auf die Verfügbarkeit und wird durch den §303b StGB (Computersabotage) bestraft. Nur die Durchführung in Bezug auf das eigene System wird als legal angesehen.
Auch Systemverwalter nutzen den Portscan als wichtiges Arbeitsmittel, um die Sicherheit des Netzwerkes zu kontrollieren. Anhand der gelieferten Ergebnisse über aktive Dienste des kontrollierten Systems kann der Verwalter nachfolgende Schritte initialisieren, um die Systemsicherheit zu verbessern [Hag15].

Da diese Bachelorarbeit den Sicherheitsaspekt thematisiert, wird der Portscan ebenfalls aus dieser Sicht betrachtet.

3.1.1 Ermittelung der IP-Adresse des S7G2s

Um an die IP-Adresse des Renesas Synergy S7G2s zu gelangen, kann man verschiedene Methoden nutzen. Eine dieser Methoden, welche für das konkrete Beispiel unter Windows durchgeführt wurde, nutzt das *„J-Link Software- und Dokumentationspaket“* (Download unter [SEG18b]) und die Datei *„SK_S7G_Heartbeat_Kit.srec“*[1] (Download unter [Ren17]).
Mit Hilfe der J-Link-Software und der srec-Datei wird das Board programmiert und man erhält als Ergebnis die in Abbildung 3.1 ersichtliche Anzeige.

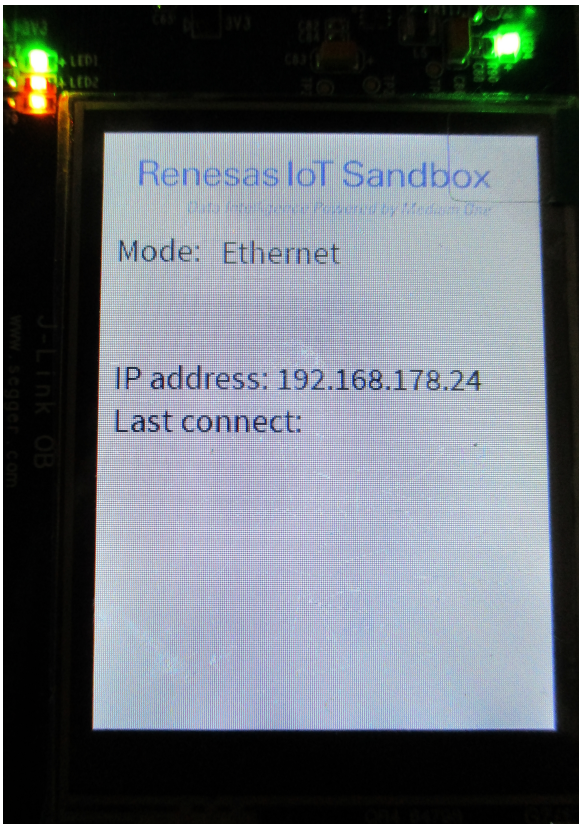

Abbildung 3.1: Ergebnis – Anzeige auf dem Board [eigene Arbeit]

Durch diese beschriebene Methode wird sowohl der Verbindungsmodus als auch die IP-Adresse des S7G2s sichtbar und verbleibt solange auf dem Display, bis ein anderes Programm auf das Board gespielt wird oder die Verbindung zu dem ausführenden Rechner abgebrochen wird.
Für nähere Informationen zur besagten Methode und deren Ablauf siehe [Ren17].

[1] srec = Das S-Record-Format ist ein ASCII-basiertes Datenformat und wird hauptsächlich verwendet, um Programmcode von Mikrocontrollern in eingebetteten Systemen abzuspeichern.

3.1.2 Durchführung eines Portscans beim Renesas Synergy S7G2

Um beim Renesas Synergy S7G2 einen Portscan durchzuführen, muss dieser zunächst über das am Jumper J11 befindliche Ethernet Interface mit dem selben Netzwerk verbunden werden, mit dem auch der ausführende Rechner, mit dessen Hilfe der Scan durchführt werden soll, verbunden ist. Zudem benötigt man die IP-Adresse des Mikrocontrollers. Wie man an diese gelangt, wird unter Punkt 3.1.1 erklärt.

3.2 JTAG und SWD

3.2.1 Joint Test Action Group

Die Abkürzung JTAG steht für „Joint Test Action Group“. Es stellt die Grundlage für den Institute of Electrical and Electronic Engineers (IEEE) Standard 1149.1 dar und ist auch bekannt unter dem Namen „Boundary-Scan“. JTAG stellt verschiedene Mittel bereit, um Hardware zu programmieren und zu debuggen, d.h., die Programme zu testen, um Fehler zu entdecken und diese anschließend beseitigen zu können [XJT18].
Zu den typischen Geräten, welche JTAG-konform sind, d.h. bei denen die Boundary-Scan-Technologie implementiert ist, gehören neben beispielsweise digitalen Signalprozessoren auch Mikroprozessoren.
Hierbei besteht der größte Vorteil darin, dass der Test eine Möglichkeit darstellt, Werte auf Pins festzulegen und zu lesen, ohne direkten physikalischen Zugriff zu haben. Aufgrund dessen wird nicht selten die Funktionalität von Mikrocontrollern dadurch getestet [JTA16].

Eine JTAG-Komponente umfasst hauptsächlich folgende Teile, welche weiterführend erklärt werden:

- Test Access Port (TAP) – auch „JTAG-Schnittstelle“ oder „JTAG-Port“ genannt
- TAP-Controller – eine Zustandsmaschine, welche die Testlogik steuert
- Zwei Schieberegister – das „Instruction Register“ (IR) und das „Data Register“ (DR)

Um bestückte Leiterplatten zu testen, benötigt man die sogenannten Test Access Port-Signale. Dafür besitzen Boundary-Scan-fähige Bausteine vier, optional fünf, zusätzliche Pins für jedes der folgenden Signale:

- TCK Test Clock – *Prüftakt:*
  Signal zur Synchronisierung der internen Zustandsmaschine. Alle anderen Signale sind zu TCK synchron, d.h., dass es umschalten muss, damit etwas passiert.

- TMS Test Mode Select – *Testmoduswahl:*
  Signal, um den nächsten Zustand des Prüftaktes zu bestimmen und den TAP-Controller zu steuern. Es wird an der steigenden Flanke des TCKs abgetastet. TMS ist parallel mit allen integrierten JTAG-Schaltungen verbunden, deshalb verändern sich alle TAP-Controller zusammen [Nic18].

- TDI Test Data Input – *Testdateneingabe:*
  Signal, um die Daten, welche in die Logik des Geräts übermittelt werden, zu repräsentieren. Dieses Signal wird, wenn sich die interne Zustandsmaschine im korrekten Zustand befindet und alle TAP-Controller synchronisiert sind, an der steigenden Flanke des TCKs abgetastet.

- TDO Test Data Output – *Testdatenausgabe:*
  Signal, um die Daten, welche aus der Logik des Geräts geleitet werden, zu repräsentieren. Es wird, anders als TDI, aber unter den gleichen Voraussetzungen, an der fallenden Flanke des TCKs abgetastet.

- TRST Test Reset – *Testzurücksetzung:*
  Der TRST-Pin ist optional und kann genutzt werden, um die Zustandsmaschine des TAP-Controllers zurückzusetzen [XJT18].

Ein typisches Board umfasst in der Regel mehrere JTAG-fähige Komponenten, wie beispielsweise Complex Programmable Logic Devices[2] (engl., kurz CPLD) oder Field Programmable Gate Arrays (engl., kurz FPGA), welche miteinander verbunden werden können.
Um den Scan durchzuführen, werden die einzelnen Bausteine in Reihe geschaltet. Dabei wird der TDO-Pin des ersten Elements mit dem TDI des darauffolgenden Bausteins verbunden. Dadurch entsteht eine serielle Verknüpfung, welche als Scankette bezeichnet wird [JTA16]. Hierbei ist es allerdings wichtig, dass sich alle TAP-Controller der Kette im gleichen Zustand befinden, was nach dem Einschalten nicht

[2]CPLDs sind komplexe programmierbare Logikbausteine, d.h. programmierbare logische Schaltungen, zu denen auch die FPGAs gehören. Sie unterscheiden sich allerdings in ihrer Komplexität.

unbedingt der Fall sein muss. Um dies zu erreichen, werden alle TAP-Controller zurückgesetzt. Selbst wenn der TRST-Pin nicht vorhanden ist, gibt es eine Möglichkeit, welche nachfolgend unter Punkt 3.2.2 erklärt wird [Nic18]
Neben den bereits genannten Signalen werden zusätzlich noch die Signale GND (Ground) und VCC (Voltage at the common Collector) benötigt, um die Stromversorgung zu gewährleisten.
Hierbei hat der VCC-Pin die Aufgabe, den integrierten Schaltkreis mit positiver Spannung zu versorgen, während GND den Massepunkt in einer elektrischen Schaltung darstellt.

In Abbildung 3.2 wird der Aufbau eines JTAG-fähigen Geräts inklusive der angelegten Signale grob dargestellt.

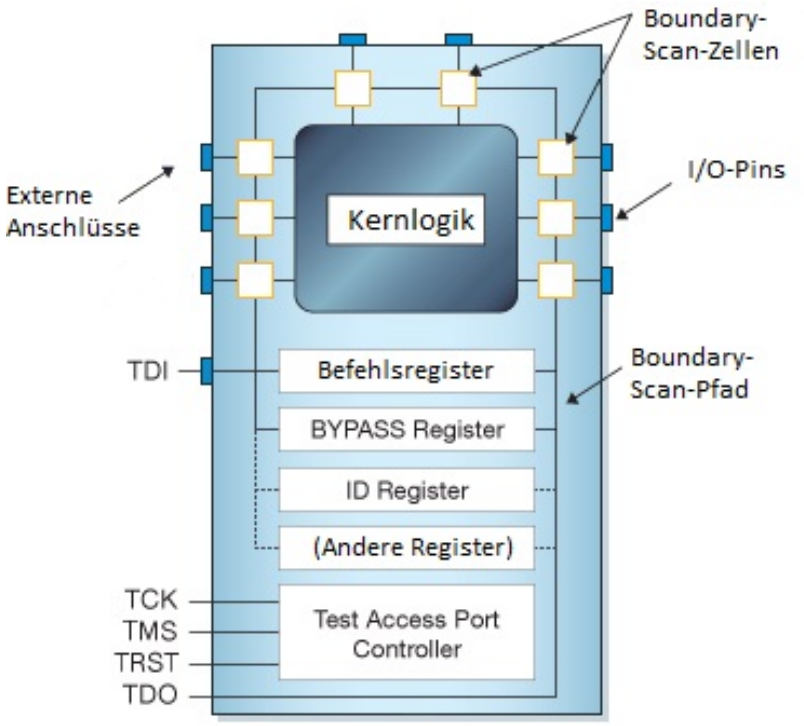

Abbildung 3.2: Schematische Darstellung eines JTAG-fähigen Geräts [eigene Arbeit, modifiziert nach [XJT18]]

Eines der wichtigsten Register, welches für den Test benötigt wird, ist das Boundary-Scan-Register (BSR). Viele Bausteine werden speziell für den JTAG-Test durch das besagte Register ergänzt.
Dieses besteht aus einer Anzahl an Boundary-Scan-Zellen, wobei sich diese zwischen den Pins und der Kernlogik, d.h. am Umfang (engl. Boundary) des Bausteins, befinden. Verbunden ist das Ganze mit einer Leiterplatte. Hierbei sind die Zellen normalerweise im Inneren und deshalb von Außen betrachtet nicht sichtbar, sie können

jedoch im Testmodus verwendet werden, um die einzelnen Werte aus den Gerätepins zu lesen oder um Einstellungen vorzunehmen.
Zudem existieren verschiedene Arten von Boundary-Zellen. Im 1149.1-Standard sind zehn Typen definiert, neben denen aber auf Wunsch der Hersteller für JTAG-fähige Geräte auch Nicht-Standard-Zellen existieren. Die Aufgabe des BSR ist es, alle Signale, die zwischen der Kernlogik und den Pins übertragen werden, seriell abzufangen.

Neben dem BSR gibt es noch weitere Register, die beim Boundary-Scan eine Rolle spielen. Grundsätzlich sind zwei Arten derer vorhanden, die mit JTAG verknüpft sind: ein Befehlsregister und zwei oder mehr Datenregister.

Befehlsregister

Befehlsregister enthalten aktuelle Anweisungen und deren Inhalt wird vom TAP-Controller genutzt. Dieser entscheidet, wie er mit empfangenen Signalen umgeht und verwendet dafür die Daten aus jenem Register. Auf diese Art und Weise wird beispielsweise auch festgelegt, an welche Datenregister die Signale weitergeleitet werden.

Datenregister

Neben dem BSR existieren zwei weitere primäre Datenregister – das IDCODES-Register und das BYPASS-Register. Alle drei Register sind primär notwendig als Teil des JTAG-Standards, während alle weiteren Datenregister optional sind.

Boundary-Scan-Register Das BSR ist das wichtigste Prüfdatenregister. Seine Aufgabe besteht darin, die Daten und Signale zu und von den I/O-Pins des Geräts zu übermitteln.

IDCODES-Register Das IDCODES-Register beinhaltet sowohl den ID-Code als auch die Revisionsnummer des Geräts, so dass dieses mit der BSDL[3]-Datei verknüpft werden kann.

BYPASS-Register Das BYPASS-Register ist ein Ein-Bit-Register, welches verwendet wird, um Informationen von TDI zu TDO zu übergeben[XJT18].

[3]BSDL = Boundary Scan Description Language, ist eine Teilmenge von VHDL (VHSIC Hardware Description Language), mit der beschrieben wird, wie JTAG in einem bestimmten Gerät implementiert ist. Damit ein Gerät JTAG-kompatibel ist, muss eine BSDL-Datei verknüpft sein. Diese Dateien stehen häufig kostenlos auf den Webseiten der Hersteller als Download zur Verfügung [XJT18].

3.2.2 Der TAP-Controller

Eines der Hauptziele während der Entwicklung des IEEE-Standards 1149.1 war es, die Anzahl der Pins des TAP-Controllers auf ein Minimum zu reduzieren, da viele integrierte Schaltungen über eine beschränkte Menge an Pins verfügen.
Der Test Access Port-Controller ist eine endliche Zustandsmaschine mit 16 verschiedenen Zuständen, deren Übergänge, wie bereits thematisiert, durch das TMS-Signal gesteuert werden. Das Verhalten des JTAG-Systems wird durch ihn geregelt. Die Funktionsweise wird durch das Zustandsdiagramm in Abbildung 3.3 dargestellt.

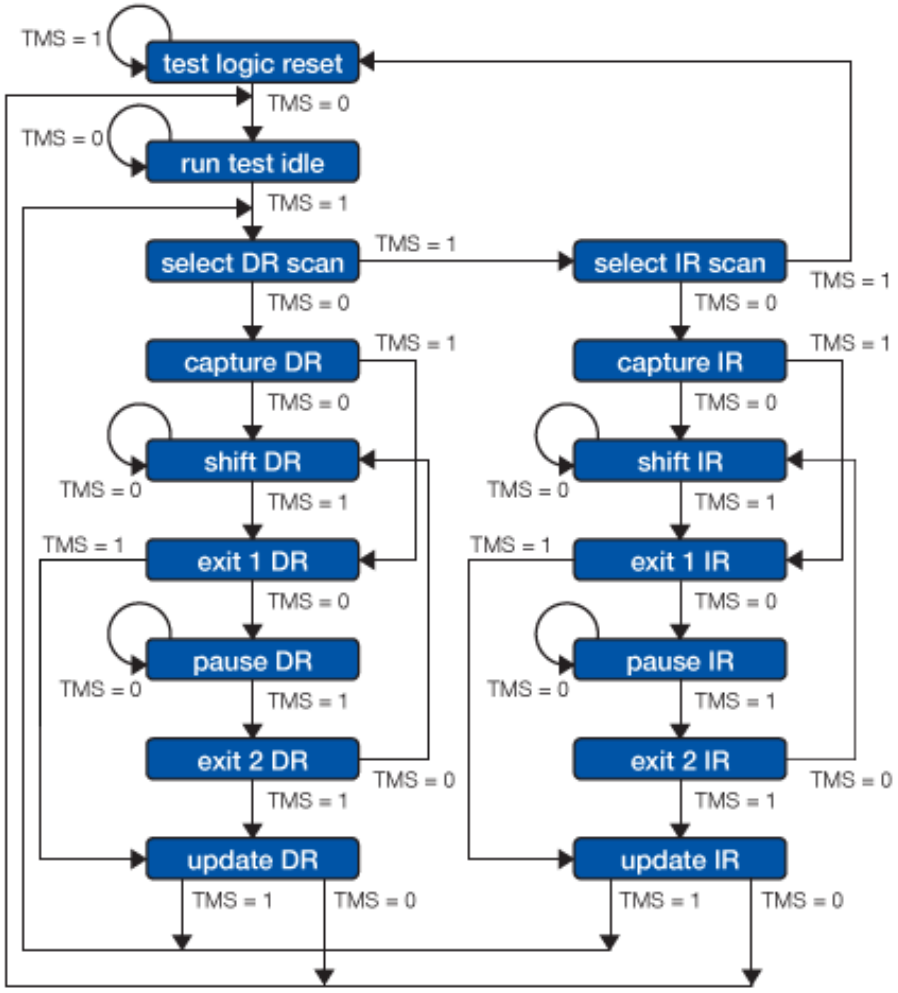

Abbildung 3.3: Zustandsdiagramm für den TAP-Controller [XJT18]

Alle Zustände haben zwei Ausgänge, so dass alle Übergänge über ein einzelnes TMS-Signal gesteuert werden können, welches auf TCK abgetastet wird.
Man beachte, dass in den Zuständen, deren Namen mit „DR“ enden, die Testdatenregister arbeiten, während in denjenigen Zuständen, deren Namen mit „IR“ enden, das Befehlsregister arbeitet.

Eine Bewegung entlang eines Zustandsübergangs tritt bei jeder ansteigenden Flanke von TCK auf. Die Nullen und Einsen, die neben den Zustandsübergängen stehen, zeigen den Wert, der zum Zeitpunkt der nächsten ansteigenden Flanke von TCK für den bestimmten Übergang an TMS vorhanden sein muss. Befindet sich ein TAP-Controller z.B. im Zustand „Select DR Scan“ und TMS ist zu diesem Zeitpunkt Null, so ändert sich der Status des Controllers in „Capture DR“, sobald TCK umgeschalten wird.

Acht der 16 Controller-Zustände bestimmen den Betrieb der Testlogik, so dass folgende Testfunktionen ausgeführt werden können:

Test Logic Reset *(Test-Logik-Zurücksetzung)* In diesem Controller-Zustand wird die gesamte Testlogik zurückgesetzt. Hierbei wird das Test Logic Reset von der systeminternen Kernlogik getrennt, wodurch der normale Betrieb des Chips ohne Störung ermöglicht wird [MT92].
Wie bereits thematisiert, müssen sich die einzelnen TAP-Controller im selben Zustand befinden, damit ein Boundary-Scan funktioniert. Zum Erreichen dieser Synchronisation gibt es bei Abwesenheit des TRST-Pins die folgende Alternative: Der Controller fällt immer wieder in den Test Logic Reset-Zustand, sobald TMS für fünf Takte auf Eins ist [Nic18]. Das bedeutet, dass unabhängig vom Startzustand des TAP-Controllers der Status des Test Logic Resets erreicht werden kann, indem TMS auf eins gehalten wird und fünf steigende Flanken an TCK angelegt werden.
Wenn ein TRST-Pin bereitgestellt wird, kann er, sowohl beim Einschalten als auch an jedem gewünschten Punkt während des Betriebs, verwendet werden, um den Controller asynchron in den Test Logic Reset-Zustand zu zwingen [MT92].

Run Test/Idle *(Lauftest/Leerlauf)* Der Betrieb der Testlogik in diesem Steuerungszustand ermöglicht die Ausführung bestimmter Operationen, welche von der aktuellen Anweisung abhängen. Ist beispielsweise der Befehl eines Selbsttests aktiviert, so wird dieser ausgeführt, wenn sich der Controller in diesem Zustand befindet.

Capture DR *(Erfassungs DR)* In diesem Zustand geht es um die Erfassung der Datenregister. Jeder Befehl muss ein oder mehrere Testdatenregister identifizieren, die im Testmodus aktiviert sind. Diese werden anschließend anhand der aktuellen Anweisung an der steigenden Flanke von TCK ausgewählt.

Shift DR *(Verschiebungs DR)* Jeder Befehl muss ein einzelnes Testdatenregister

identifizieren, das in diesem Zustand zum Übermitteln von Daten zwischen TDI und TDO verwendet werden soll. Shifting ermöglicht es, die zuvor erfassten Daten zu prüfen und neue Test-Eingabedaten einzufügen und übermittelt die Daten im aktuell ausgewählten Register.
Die Verschiebung tritt während des Steuerungszustandes bei der steigenden Flanke von TCK auf. Besonders ist, dass der Ausgang TDO lediglich in diesem, und dem Shift IR-Status aktiv ist (in allen anderen Controller-Zuständen ist er inaktiv).

Update DR *(Aktualisierungs DR)* Dieser Controller-Status kennzeichnet den Abschluss des Verschiebevorgangs. Einige Testdatenregister können mit einem Ausgang versehen sein, welcher Ergebnisse parallel zwischenspeichert. Dieser soll verhindern, dass Signale gestört werden, während neue Daten in das Register verschoben werden.
Werden solche Testdatenregister durch eine aktuelle Anweisung ausgewählt, werden die neuen Daten in diesem Steuerungszustand an der fallenden Flanke von TCK zu ihren parallelen Ausgängen übertragen.

Capture, Shift und Update IR Diese Controller-Zustände sind analog zu Capture DR, Shift DR bzw. Update DR, bewirken allerdings den Betrieb des Befehlsregisters.
Durch Eingabe dieser Zustände kann ein neuer Befehl eingegeben und an die Testdatenregister und/oder andere spezielle Schaltungen angelegt werden. Dieser Befehl wird an der fallenden Flanke von TCK im Update IR-Zustand des Controllers gegenwärtig.

Die Funktionen der Befehls- und Prüfdatenregister in jedem dieser Steuerungszustände werden in dem folgenden Abschnitt ausführlicher beschrieben.

In den übrigen acht Controller-Zuständen führt die Testlogik keine Operationen aus. Die Pause-Zustände (Pause DR und Pause IR) werden genutzt, um Schaltprozesse vorübergehend anzuhalten, zum Beispiel während ein Gerät, welches die Testlogik steuert, mehrere Testdaten aus dem Backup-Speicher (z. B. Disk) holt.

Die letzten sechs Controller-Zustände Select DR Scan, Select IR Scan, Exit1 DR, Exit1 IR, Exit2 DR und Exit2 IR sind Entscheidungspunkte, welche eine Auswahl hinsichtlich der zu befolgenden Route ermöglichen. Zum Beispiel wird im Exit1 DR-Controller-Zustand abhängig von dem an der TMS-Eingabe angelegten Signal eine Wahl getroffen zwischen dem Eintritt in den Pause DR-Zustand oder in den Update DR-Zustand. Ohne den TAP-Controller müssten die neun Funktionen, die von den

zuvor beschriebenen Zuständen (Test Logic Reset, Run Test und Idle sowie Capture, Shift und Update für die beiden Registertypen) erfüllt werden, mit mindestens vier einzelnen Eingaben ausgewählt werden. Mit dem TAP Controller hingegen wird nur ein Steuereingang (TMS) benötigt.
Die Nachteile dabei sind allerdings, dass ausreichend Logik in jede Komponente eingebaut werden muss, um die an TMS empfangenen Signale dekodieren zu können. Außerdem ist die Fähigkeit, zwischen den Funktionen zu wechseln, leicht eingeschränkt, da beliebige Zustandsänderungen nicht immer möglich sind [MT92].

3.2.3 Serial Wire Debug

Serial Wire Debug (kurz SWD) ist eine alternative Schnittstelle für ein serielles JTAG-Protokoll. Es verwendet allerdings, anders als JTAG, nur zwei Pins: SWCLK und SWDIO. Hierbei stellt SWDIO (engl. Serial Wire Data Input/Output) den einzigen Datenpin dar, welcher bidirektional funktioniert und somit Signale in beide Richtungen übertragen kann. Bei SWCLK (engl. Serial Wire Clock) handelt es sich um das Taktsignal zur CPU.
Um mit einem Gerät über SWD zu kommunizieren, werden die vorhandenen Daten synchron an den I/O-Port gesendet. Mit jeder steigenden Flanke von SWCLK wird ein Bit Daten an den SWDIO-Pin gesendet oder von ihm empfangen. Sowohl JTAG als auch SWD können über den gleichen Anschluss verwendet werden.
Wie auch bei JTAG existiert zudem ein optionaler Pin: SWO (engl., Serial Wire Output), welcher als zusätzlicher Ausgabe-Port verwendet werden kann. Dieser liegt normalerweise beim TDO-Anschluss an und ermöglicht der CPU die Ausgabe benutzerdefinierter Daten asynchron zu SWCLK [SEG15].
Einer der Vorteile dieser Debugging-Methode gegenüber JTAG ist, dass wenig Strom verbraucht wird und zudem keine zusätzlichen VCC- oder GND-Pins benötigt werden [ARM18b].

3.2.4 Anwendung von JTAG und SWD auf dem Renesas Synergy S7G2

Ziel der nachfolgenden Methoden ist es, herauszustellen, ob und wie die Anwendung von JTAG auf dem Renesas Synergy S2G2 Mikrocontroller in der Praxis möglich ist. Dabei bietet der S7G2 mehrere Alternativen für das JTAG-Debugging. Einerseits ist ein integrierter SEGGER J-Link JTAG-Debugger über den DEBUG_USB-Anschluss J19 zugänglich. Andererseits besteht über den Anschluss J18 auch direkter Zugriff auf den JTAG/SWD-Port des S7G2 Mikrocontrollers. Sowohl für den indirekten als auch für den direkten Zugriff benötigt man das *J-Link Software- und Dokumentationspaket*.

Beide Varianten werden im Folgenden näher erläutert.

3.2.4.1 Indirekter Zugriff

Benötigt wird für den indirekten Zugang das unter Punkt 3.1.1 bereits erwähnte *J-Link Software- und Dokumentationspaket*. Dieses enthält viele verschiedene ausführbare Programme, von denen lediglich eins im Folgenden näher betrachtet wird. Zudem ist es erforderlich, dass das Board mittels USB-Kabel über den DEBUG_ USB-Anschluss J19 mit dem ausführenden Rechner verbunden ist.

Das J-Link-Paket bietet folgende Methode für den indirekten Zugriff:

„JFlash.exe“

Um JFlash zu verwenden, ist es zu Beginn notwendig, ein neues Projekt zu erstellen. Dafür muss das Zielgerät, in diesem Fall `R7FS7G27H`, ausgewählt und als Zielschnittstelle *SWD* eingestellt werden. Die vorgegebenen Einstellungen für die Bytereihenfolge, hier *Little Endian*, und die Frequenz, hier *4000 kHz*, können beibehalten werden. Nachdem so das neue Projekt vorbereitet ist, kann anschließend die Verbindung zum Zielgerät hergestellt werden, indem man in der Menüleiste folgende Punkte auswählt: `Target` → `Connect`. Danach erfolgt im Log-Feld die Bestätigung der Verbindung, quittiert durch „Connected successfully“.
Um nun die gespeicherten Daten des gesamten Chips auszulesen, wählt man in der Menüleiste nacheinander die Punkte `Target` → `Manual Programming` → `Read Back` → `Entire Chip` aus.

Eine weitere Methode, um den Speicher des S7Gs über SWD und mit Hilfe des Softwarepakets auszulesen, nutzt die Datei *„JMem.exe“*. Allerdings bietet diese Vor-

gehensweise im Vergleich zu der obigen weniger Funktionen bei gleichem Ergebnis, so dass sie hier nicht näher betrachtet wird.

3.2.4.2 Direkter Zugriff

Neben dem indirekten Zugang kann auf JTAG auch direkt über den Anschluss J18 mit Hilfe eines externen Debuggers, welcher ebenfalls von der Firma SEGGER ist, zugegriffen werden (ersichtlich in Abbildung 3.6).
Um den Direktanschluss zu aktivieren, müssen die Widerstände (R107, R108, R109 und R110), die den integrierten J-Link-Debugger mit der MCU verbinden, entfernt werden. Diese befinden sich oberhalb des direkten JTAG-Pins, zwischen dem Jumper J18 und dem LCD-Display.
Anschließend wird der S7G2 mit dem externen SEGGER J-Link Debugger, anhand der vorgegeben JTAG-Pinbelegungen des Debuggers und der Pinzuweisung des Mikrocontrollers, per Kabel verbunden.

Die genaue Belegung der Pins des S7G2 Boards und der dort jeweils anliegenden Signale des Direktanschlusses J18 wird in der folgenden Abbildung dargestellt.

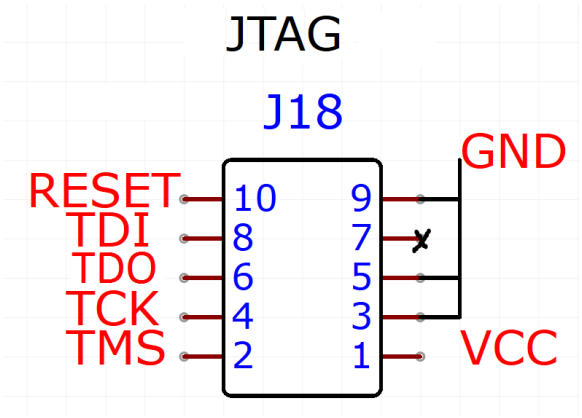

Abbildung 3.4: Pinbelegung für JTAG am Jumper J18 [eigene Arbeit, modifiziert nach [Ren15]]

Die Zuordnung der Pins des SEGGER Debuggers und der jeweiligen Signale wird in der Abbildung 3.5 dargestellt.

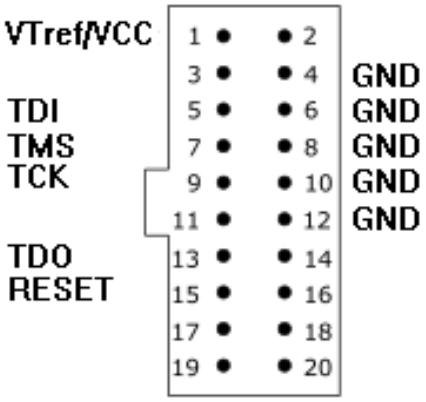

Abbildung 3.5: Pinbelegung des externen SEGGER J-Link Debuggers für JTAG [eigene Arbeit, modifiziert nach [SEG96]]

Da die Abstände der einzelnen Buchsen des Direktanschlusses auf dem Board sehr gering sind, wurden zwei Stiftleisten, mit einem Abstand von 1,27 mm zwischen den einzelnen Pins, am Jumper J18 angelötet und zusätzlich, um einen besseren Halt zu gewährleisten, Kabel genutzt, welche an einem Ende über Abgreifklemmen verfügen. Diese Klemmen werden nun an den jeweiligen Steckern angebracht und die anderen Enden der Kabel mit den zugehörigen Pins des SEGGER Debuggers verbunden. Abschließend wird dieser mit Hilfe des USB-Kabels mit dem ausführenden Rechner verbunden.

Der daraus resultierende Versuchsaufbau wird in der folgenden Abbildung dargestellt.

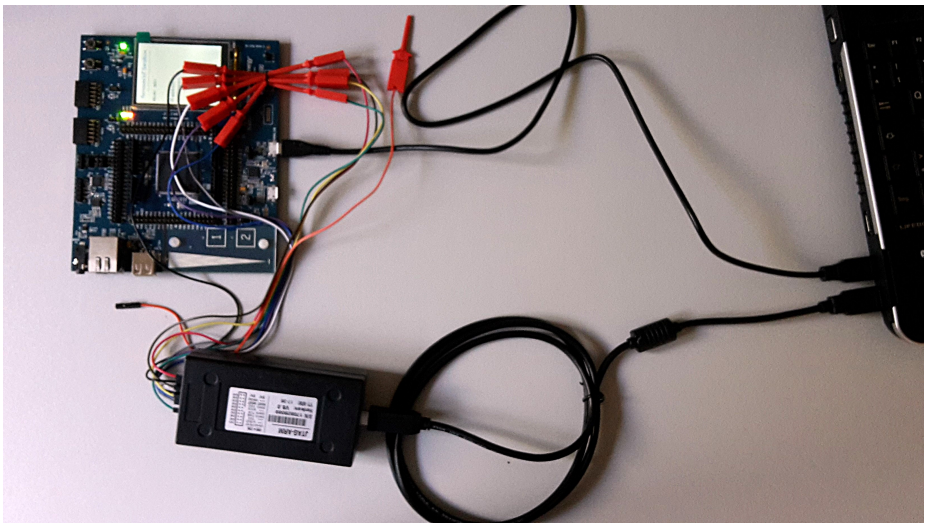

Abbildung 3.6: Versuchsaufbau für JTAG-Anwendung über direkten Zugriff am Jumper J18 [eigene Arbeit]

Um den direkten Zugriff zu nutzen, wird der GDB Server benötigt, welcher zunächst erklärt wird.

Die Abkürzung GDB steht für GNU Debugger. Hierbei ist GNU ein Projekt, welches 1984 von der Free Software Foundation (FSF) aufgebaut wurde. Ziel dessen war die Entwicklung eines lizenzfreien Betriebssystems, welches dem Unix ähnlich sein sollte, sich davon aber dennoch klar abgrenzen sollte [Sta00].
Der GDB Server ist ein Steuerungsprogramm für Unix-ähnliche Systeme, welches ein *Remote Debugging* anderer Programme ermöglicht, d.h., er erlaubt dem GNU Debugger, sich von einer anderen Einheit aus zu verbinden. Hierbei ist es lediglich notwendig, dass sich die ausführbare Datei, welche debuggt werden soll, auf dem Zielsystem befindet. Eine Kopie der Debug-Binärdatei und der Quellcode befinden sich unterdessen auf dem Computer des Entwicklers („Host“). Um die Verbindung herzustellen, kann entweder TCP oder eine serielle Leitung verwendet werden [Sta+02].

Außerdem wird ebenfalls das *J-Link Software- und Dokumentationspaket* benötigt, und zusätzlich dazu die *„GNU Arm Embedded Toolchain“* (Download unter [ARM18a]).

Das J-Link-Paket bietet hierbei folgende Dateien für den direkten Zugriff:

„JLinkGDBServer.exe“

Der entfernte Zielhost wird mit Hilfe des J-Link GDB Servers in Erfahrung gebracht. Um diesen für den direkten Zugriff nutzen zu können, ist es zu Beginn notwendig, wie bei der indirekten Methode, die Einstellungen anhand des zu testenden Geräts anzupassen.
Hierfür muss das Zielgerät, in diesem Fall `R7FS7G27H`, ausgewählt und als Zielschnittstelle *JTAG* eingestellt werden. Die vorgegebene Einstellung für die Frequenz unter dem Punkt „fixed“, hier *4000 kHz*, kann beibehalten werden. Anhand der Gerätebezeichnung kann die Verbindung über die serielle Leitung, hier USB, hergestellt werden.

Zudem bietet die *„GNU Arm Embedded Toolchain“* im Ordner */gcc_toolchain/bin* folgende Datei für den direkten Zugriff:

„arm-none-eabi-gdb.exe“

Durch das Ausführen der Datei *„arm-none-eabi-gdb.exe“* wird über die dedizierte Kommandozeile der Zugriff auf den GDB Server ermöglicht. Hierbei können durch die Eingabe verschiedener Befehle, von denen wichtige in [Hai07] aufgeführt werden,

viele Informationen gewonnen werden – sowohl über das zugrunde liegende Gerät als auch über dessen Speicher.
Um die Verbindung zum Zielgerät herstellen und Daten auslesen zu können, wird der TCP/IP Port in Verbindung mit bestimmten Eingabeaufforderungen benötigt. Diese werden in den Ergebnissen unter Punkt 4.2.2 benannt und näher erläutert.

4 Ergebnisse

Neben den Ergebnissen der praktisch durchgeführten Methoden werden in diesem Kapitel, unter Punkt 4.3, zusätzliche Schutzfunktionen des Renesas Synergy S7G2 Mikrocontrollers, wie beispielsweise Registerschreibschutz, theoretisch betrachtet.

4.1 Portscan

Im Beispiel wurde der Portscanner *nmap* verwendet. Des Weiteren wurden im Vorfeld keine Veränderungen am Controller getätigt, so dass sich die Ergebnisse auf die Standardeinstellungen des S7G2s beziehen.
Ausgeführt wurde der Portscan durch den einfachen Befehl `nmap <target IP>`, welcher die am häufigsten verwendeten 1000 TCP-Ports auf dem Zielsystem mit Hilfe der IP-Adresse scannt und jeden Port nach seinem Zustand klassifiziert – offen, geschlossen oder verdeckt.
Im Beispiel, welches in der Abbildung 4.1 dargestellt wird, wird der Portscan durch den Befehl `nmap 192.168.178.24` gestartet und liefert folgende Ergebnisse:
Zu Beginn werden 999 Ports als geschlossen klassifiziert, welche deshalb nicht explizit angezeigt werden.
Weiterführend werden die betreffenden Ports, in diesem speziellen Fall nur einer, anhand der Art und des Zustands des Ports und anhand des verwendeten Service aufgelistet und die MAC-Adresse des Gerätes angezeigt.
Wie in Abbildung 4.1 ersichtlich wird, liegt der offene TCP-Port 80 vor, welcher für den Service HTTP (engl. HTTP – Hypertext Transfer Protocol) verwendet wird. Dies ist ein Internetkommunikationsprotokoll und wird zur Übertragung von Daten auf der Anwendungsschicht genutzt, um in einen Webbrowser Internetseiten aus dem World Wide Web (kurz WWW) zu laden. Das ausführende System sendet und empfängt hierbei über den Port 80 Nachrichten vom Webserver, während Webserver- und Client kommunizieren, und wird außerdem verwendet, um HTML-Seiten oder Daten zu senden oder zu empfangen.

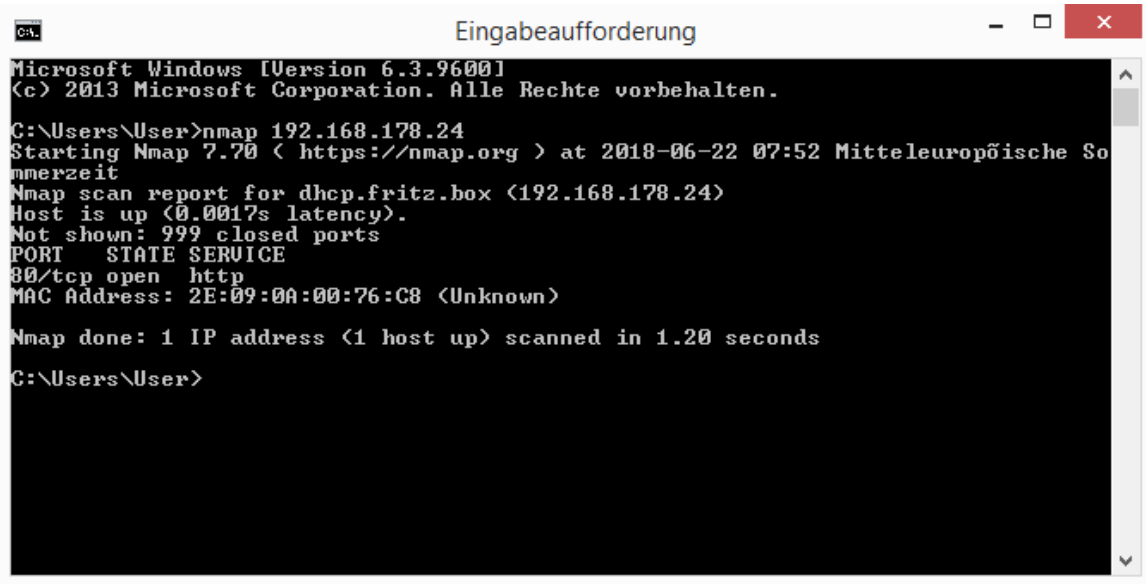

Abbildung 4.1: Ergebnis des Portscans beim Renesas Synergy S7G2 [eigene Arbeit]

Der Mikrocontroller ist über Ethernet am Jumper J11 mit dem Netzwerk und über USB am Jumper J19 mit dem auszuführenden Rechner verbunden. Hierbei muss Port 80 geöffnet sein, da der Controller als Webserver fungiert und über HTTP die Webseite zur Konfiguration, wie in Abbildung 4.2 ersichtlich, bereitstellt.

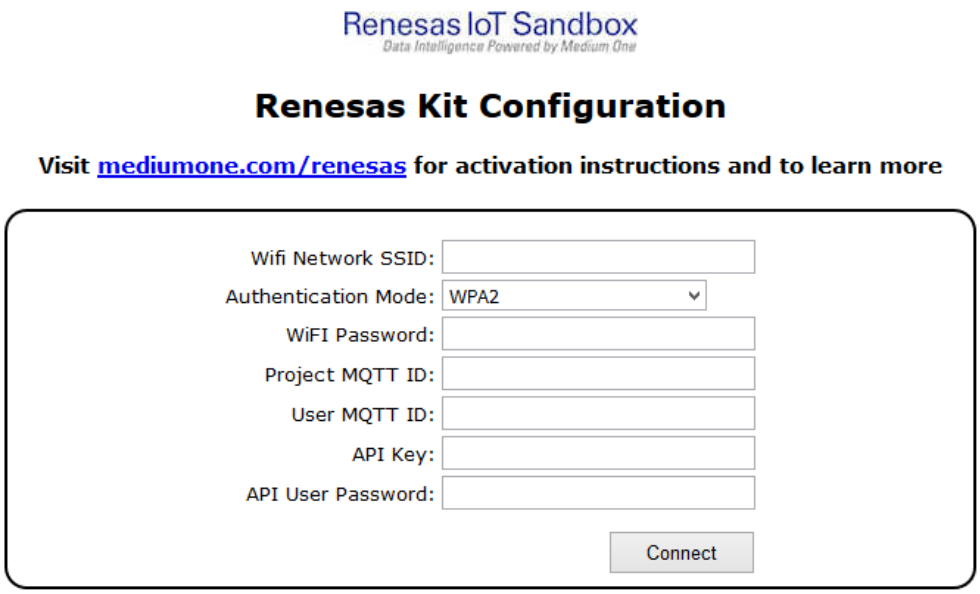

Abbildung 4.2: Kit Konfigurationsseite [eigene Arbeit]

Da der Portscan des S7G2s offen legt, dass lediglich ein Port geöffnet ist, welcher in der betreffenden Situation auch gebraucht wird, ist das Ergebnis zufriedenstellend. Es sind nur benötigte Ports offen, alle anderen sind geschlossen. Dadurch werden unnötige Angriffspunkte durch weitere offene Ports ausgeschlossen.

4.2 JTAG und SWD

Wie bereits unter 3.2.4 erwähnt, soll heraus gefunden werden, ob JTAG am Renesas S7G2 möglich ist. Für die Nutzung des direkten JTAG-Interfaces ist es hierbei notwendig, die oberhalb des Jumpers J18 befindlichen Widerstände (R107, R108, R109, R110), abzulöten. Hierbei ist zu beachten, dass anschließend der indirekte Zugriff dadurch nicht mehr verwendbar ist.
Da man durch JTAG direkten Hardware-Zugriff erlangt, eignet sich die Schnittstelle gut, um Sicherheitsmaßnahmen zu umgehen. Beispielsweise wird sie häufig benutzt, um nähere Informationen über den Bootprozess eines Systems zu bekommen. Nachfolgend sind Angreifer in der Lage, jenen zum Beispiel zu unterbrechen und Schadcode zu injizieren. Dadurch können sie das System kontrollieren und steuern. Aufgrund dessen ist es sehr wichtig, die JTAG-Schnittstelle vor eventuellen Angriffen zu schützen.

4.2.1 Indirekter Zugriff

„JFlash.exe“

Sobald das Auslesen des Speichers durch die Read-Back-Methode gestartet wird und die Statusanzeige erscheint, wird das Display grau und die vorherige Anzeige erlischt. Dies passiert, da durch den Start des Auslesens ein Reset des Controllers erfolgt, was bei dieser Methode standardmäßig stattfindet.
Nachdem die unter Punkt 3.2.4.1 aufgeführten Schritte ausgeführt wurden und durch *JFlash.exe* der komplette Chip ausgelesen wurde, erscheint folgendes Ergebnis:

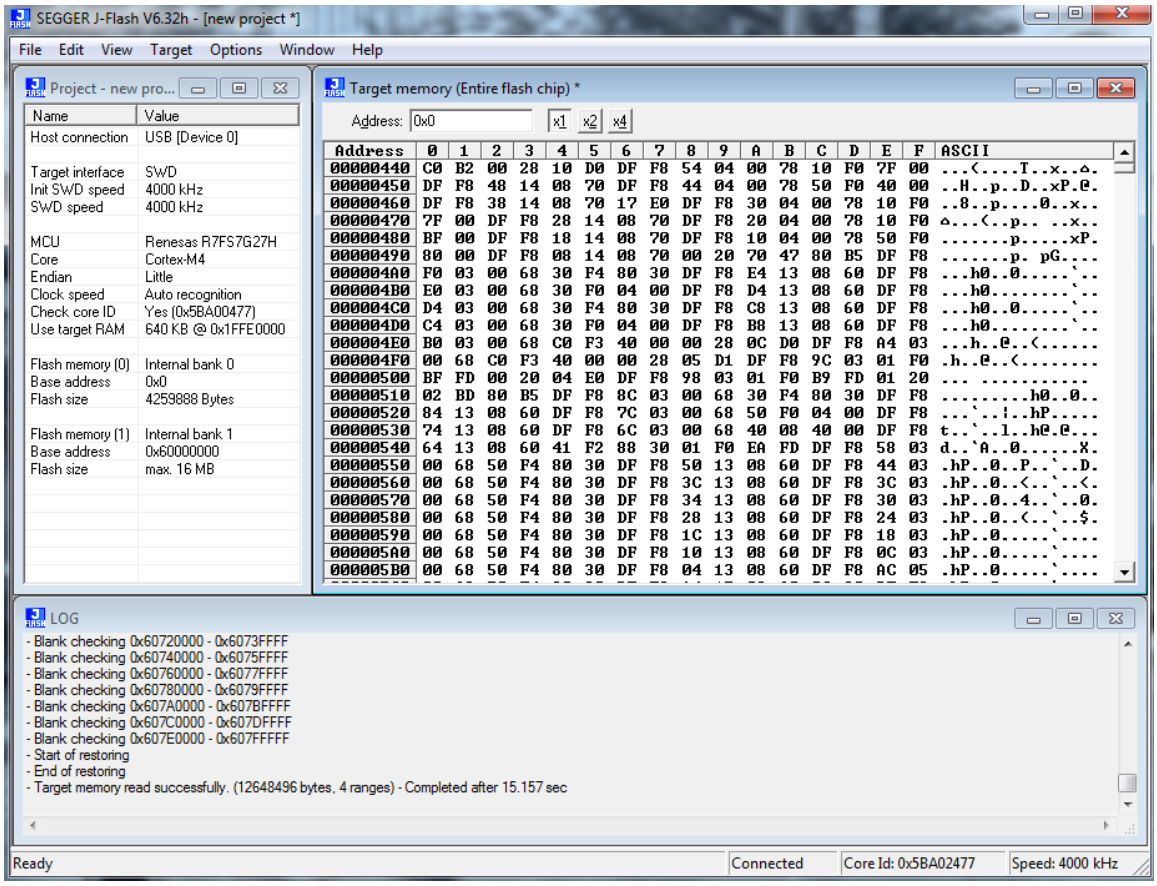

Abbildung 4.3: Ergebnis des Speicherauslesevorgangs mit *„JFlash“* [eigene Arbeit]

Sowohl auf der rechten Seite der Abbildung 4.3 als auch im Log-Feld wird das erfolgreiche Auslesen angezeigt. Zum einen wird im Log deutlich, wie viele Bereiche und Bytes ausgelesen wurden, zum anderen werden die Daten innerhalb des Speichers offen gelegt. Dadurch wird sichtbar, dass es möglich ist, durch diese Methode den Speicher des Renesas Synergy S7G2 Mikrocontrollers auszulesen.
Außerdem können durch die Projektanzeige linker Hand, in der die genauen Adressen der einzelnen Speicher angegeben sind, und durch die Zeitangabe und die Größenangabe im Log-Feld, Zusatzinformationen gesammelt werden. Weiterhin interessant ist die automatische Anzeige der ID des Kerns. Diese kann für eventuelle spätere Angriffe genutzt werden.

4.2.2 Direkter Zugriff

„JLinkGDBServer.exe“

Nachdem die unter Punkt 3.2.4.2 aufgeführten Schritte ausgeführt und die Verbindung zwischen dem GNU Debugger und dem Renesas Synergy S7G2 Mikrocontroller

mit Hilfe der Gerätebezeichnung über die USB-Schnittstelle hergestellt wurde, erscheint das in der Abbildung 4.4 dargestellte Ergebnis:

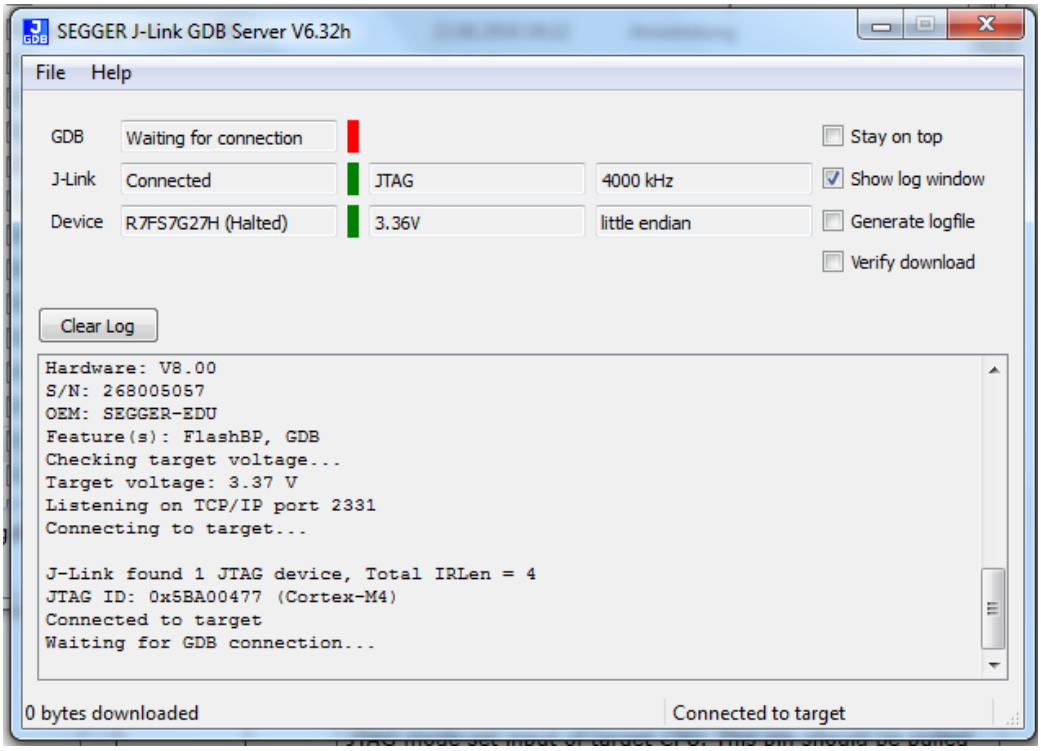

Abbildung 4.4: Anwendung von JTAG über direkten Zugriff an Jumper J18 mittels J-Link GDB Server [eigene Arbeit]

Sowohl im oberen Bereich der Abbildung 4.4 als auch im Log-Feld wird die erfolgreiche JTAG-Verbindung zum zuvor ausgewählten Gerät sichtbar. Ebenso werden alle weiteren eingestellten Parameter, wie etwa die Frequenz oder die Bytereihenfolge, angezeigt. Im Log wird zum einen deutlich, dass ein JTAG-fähiges Gerät mit der ID 0x5BA00477 und einem Cortex-M4-Prozessor erkannt wurde, zum anderen wird der TCP/IP Port 2331 offen gelegt. Dadurch wird sichtbar, dass es möglich ist, auf diese Weise eine JTAG-Verbindung zum Renesas Synergy S7G2 aufzubauen.
Die Information über den entfernten Rechner ist entscheidend für die nachfolgende Methode, denn durch die Nutzung des Ports kann anschließend der GDB Server gestartet werden.

„arm-none-eabi-gdb.exe“

Nachdem die Information über den TCP/IP Port durch den GNU Debugger bereitgestellt wurde, wird diese anschließend mit Hilfe der Toolchain und der ausführbaren *„arm-none-eabi-gdb.exe“*-Datei für die Nutzung des GDB Servers verwendet.
Durch das Öffnen der besagten Datei wird ein Terminal sichtbar, welches Zugriff auf

den GDB Server ermöglicht und in der nachfolgenden Abbildung dargestellt wird.

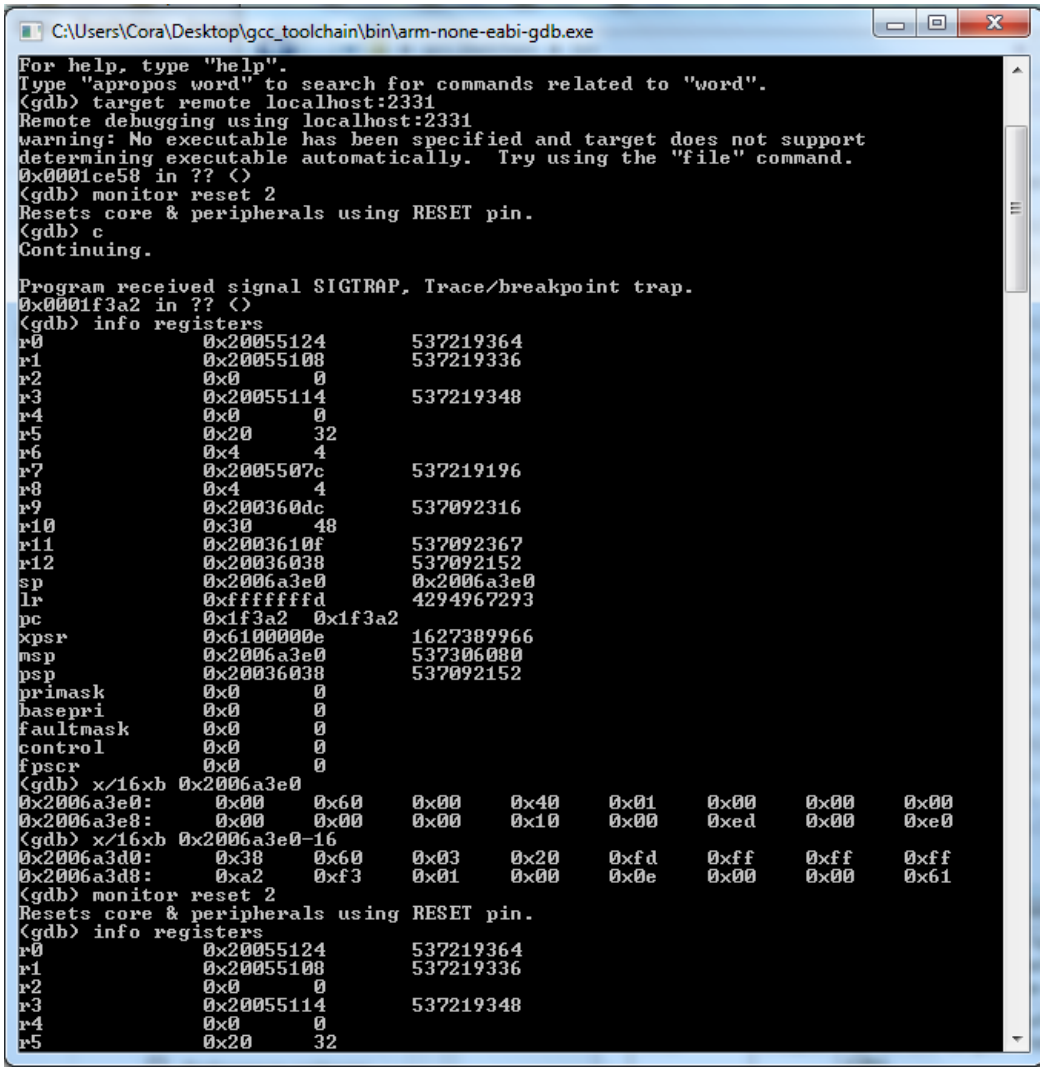

Abbildung 4.5: Anwendung von JTAG über direkten Zugriff an Jumper J18 mittels *ARM Toolchain* [eigene Arbeit]

Anhand des Befehls `target remote localhost:2331` wird die Verbindung zum Zielsystem aufgebaut. Gleich darunter wird das Remote Debugging des angegebenen Hosts bestätigt. Nach dem erfolgreichen Verbindungsaufbau ist es möglich, den Renesas Synergy S7G2 über die Kommandozeile zu steuern. Durch den Befehl `monitor reset 2` wird der Bildschirm zurückgesetzt und färbt sich dadurch grau. Abgebrochen wird dieser Reset durch die Eingabe von `c`, welches für „continue“ (fortsetzen) steht und die vorherige Bildschirmanzeige als Ergebnis hat.
Weiterführend werden nach dem Ausführen des Befehls `info registers` die verschiedenen Register des Mikrocontrollers mit deren Startadressen und Inhalten sichtbar. Um sich die ausgelesenen Speicherinhalte konkret auszugeben, kann das Kommando `x/16xb` *<Adressangabe>* verwendet werden. Im konkreten Beispiel werden mit Hilfe

der Eingaben `x/16xb 0x2006a3e0` und `x/16xb 0x2006a3e0-16` sowohl die aktuellen als auch die vorherigen Speicherinhalte des Stack Pointers sichtbar gemacht.

Anhand der Nutzung der genannten Befehle und der daraus resultierenden Ergebnisse wird sichtbar, dass es möglich ist, durch diese Methoden und mit Hilfe der direkten JTAG-Schnittstelle den Speicher des Renesas Synergy S7G2 Mikrocontrollers auszulesen.

4.3 Theoretische Betrachtung vorhandener Schutzfunktionen

4.3.1 Schreibschutz für Register

Für den Renesas S7G2 Mikrocontroller gibt es eine Funktion, um wichtige Register vor unbefugtem Schreibzugriff zu schützen. Die zu schützenden Register werden hierbei mit Hilfe des Protect-Registers (PRCR) festgelegt, welches verschiedene Protect-Bits enthält. Der Aufbau dessen wird im Anhang unter der Abbildung A.1 veranschaulicht. Je nachdem, ob das Protect-Bit auf Null oder auf Eins gesetzt ist, wird der Schreibzugriff für das jeweilige Register deaktiviert bzw. aktiviert.
Dabei existieren unterschiedlich positionierte PRC-Bits, wobei die Position gleich Null, Eins oder Drei sein kann. Diese aktivieren oder deaktivieren den Schreibvorgang für die folgenden Register:

PRC-Bit 0 Betrifft alle Register, die sich auf die Clock Generetaion Circuit (engl., Taktgenerierungsschaltung) beziehen.

PRC-Bit 1 Betrifft alle Register, welche sich auf die Energiesparmodi und die Batterie-Backup-Funktion beziehen.

PRC-Bit 3 Betrifft alle Register, welche für die Niederspannungserkennung[1] genutzt werden [Ren18a].

[1] Das Low Voltage Detection-Modul überwacht den Spannungspegel, der in den VCC-Pin eingegeben wird und erzeugt ein Reset, sobald die Versorgungsspannung zu niedrig ist [Ren18a].

4.3.2 Memory Protection Unit

Eine Memory Protection Unit (kurz MPU) stellt eine Computerhardwareeinheit dar, welche einen Speicherschutz bereitstellt.
Sie wird normalerweise als Teil der Zentraleinheit (CPU) implementiert. MPU ist eine funktional reduzierte Version der Speicherverwaltungseinheit (Memory Management Unit, kurz MMU), die lediglich Speicherschutz unterstützt und demnach keine Speicherverwaltungsaufgaben übernimmt. Dieser Baustein wird normalerweise in Low-Power-Prozessoren implementiert, die nur Speicherschutz benötigen und nicht die vollständigen Funktionen einer Speicherverwaltungseinheit, wie beispielsweise virtuelle Speicherverwaltung [Waw09].
Die MPU ermöglicht es der privilegierten Software, Speicherbereiche zu definieren und ihnen Speicherzugriffsberechtigungen und Speicherattribute zuzuweisen. Hierbei variiert die Anzahl der unterstützten Speicherbereiche in Abhängigkeit von der Implementierung des Prozessors. Die MPU überwacht Transaktionen, einschließlich der Befehlsabrufe und Datenzugriffe vom Prozessor. Diese können eine Fehlerausnahme auslösen, wenn eine Zugriffsverletzung erkannt wird.
Der Hauptzweck des Speicherschutzes besteht darin, zu verhindern, dass ein Prozess auf einen bestimmten Speicherbereich zugreifen kann, welcher ihm nicht zugewiesen wurde. Dies soll verhindern, dass entweder ein Fehler im Programm oder Malware andere Prozesse oder das Betriebssystem selbst beeinflussen kann [Sti12].

4.3.2.1 MPUs des Renesas Synergy S7G2s

Im Folgenden soll es um die MPUs gehen, welche im ARM Cortex-M4 Prozessor integriert sind, da der Renesas Synergy S7G2 Mikrocontroller über jenen verfügt.

Mit Hilfe der MPU kann man sowohl Privilegienregeln als auch Zugriffsregeln für den Arbeitsspeicher erzwingen und Prozesse trennen. Die MPU ist hierbei eine optionale Komponente und unterstützt das ARMv7 Protected Memory System Architecture-Standardmodell. Die geschützte Systemarchitektur ist eine optionale Funktion auf Systemebene zur Unterstützung eines Benutzer- (nicht privilegiert) und Supervisor- (privilegierten) Softwaremodells, und um die Zugriffsrechte zu steuern [ARM10b].
Die Arm Cortex-M4 MCU bietet zwei Features in Bezug auf illegalen Speicherzugriff. Zum einen verfügt die CPU über eine Standardspeicherzuordnung. Zum anderen verfügt die MCU im Bereich des Speicherschutzes sowohl über drei Memory Protection Units als auch über eine CPU Stack Pointer (SP) Monitor-Funktion [Ren18a].

Die verschiedenen Spezifikationen werden im Folgenden näher beschrieben.

4.3.2.2 CPU Stack Pointer Monitor

Der CPU Stack Pointer Monitor überwacht den Stack Pointer. Er erkennt sowohl Stack Overflows als auch Stack Underflows. Grundsätzlich verfügt die dynamische Datenstruktur über zwei Operationen, um auf den Speicherstapel zuzugreifen. Zum einen PUSH, um ein Element hinzuzufügen und zum anderen POP, um das zuletzt hinzugefügte Element zu entfernen.
Angenommen, der Stack ist leer und durch POP soll das letzte Element entfernt werden, so entsteht ein Fehler, welcher als Stack Underflow bezeichnet wird. Dies ist der Monitor in der Lage zu detektieren. Ebenso erkennt er, wenn zu große Datenmengen in einen dafür zu kleinen Bereich im Speicher geschrieben werden sollen.
Durch einen Stack Overflow kommt es zum Überschreiben von bestimmten Stellen im Ziel-Speicherbereich. Aufgrund dessen stellen die besagten Stapelüberläufe laut [SKG18] eine der „schwerwiegendsten Sicherheitsbedrohungen in der IT“ dar. Ein Angreifer kann sich auf diese Art und Weise Zugriffsrechte beschaffen – beispielsweise durch das Ausführen von eigenem Code im RAM des Zielsystems oder die Installation von Viren – und als Folge davon die Steuerung des gesamten Systems übernehmen.
Deshalb ist es sehr wichtig, den Speicher gegen solche Angriffe zu schützen. Es spricht demnach für die Sicherheit des S7G2s, dass ein solcher Speicherschutz durch den Stack Pointer Monitor vorliegt.

Da die CPU über zwei Stack Pointer verfügt, unterstützt sie zwei SP Monitore. Vorhanden sind hierbei der Main Stack Pointer (MSP) und der Process Stack Pointer (PSP). Sobald ein Stack Over- oder Underflow erkannt wird, wird durch den SP Monitor ein Reset oder ein nicht maskierbarer Interrupt[2] erzeugt.
Um den SP Monitor zu aktivieren, wird das Enable-Bit (engl. „aktivieren“) des selben im Zugriffssteuerungsregister (Stack Pointer Monitor Access Control Register – MSPMPUCTL und PSPMPUCTL) auf Eins gesetzt. Weiterführend ist es möglich, die Register, die sich auf den CPU Stack Pointer Monitor beziehen, zusätzlich zu schützen, indem man das zugehörige Protect-Bit auf Eins setzt [Ren18a].

4.3.2.3 Arten von MPUs beim Renesas Synergy S7G2

Um die verschiedenen MPU-Arten betrachten zu können, wird zuvor das Master-Slave-Prinzip näher erläutert.

[2]Interrupts sind Programmunterbrechungen und liegen in zwei Arten vor. Während ein maskierbarer Interrupt zugelassen werden kann, aber nicht erlaubt werden muss, kann ein nicht maskierbarer Interrupt im Vergleich dazu nicht gesperrt werden.

Das Master-Slave-Prinzip ist eine Methode, um Aufgaben innerhalb eines Systems hierarchisch zu verteilen und zu organisieren. Hierbei ist der Master übergeordnet und steuert die untergeordneten Recheneinheiten – die Slaves. Dafür verfügt lediglich der Master über den uneingeschränkten Zugriff auf bestimmte Ressourcen, während die Slaves keinen direkten Zugang darauf haben. Dies bedeutet, dass der Master nicht nur Aufgaben und die Kommunikation, sondern auch Speicherressourcen kontrolliert, steuert und zuteilt [ITW14].

ARM MPU Die ARM MPU ist in der CPU integriert und stellt acht Regionen mit Subregionen und Hintergrundregionen zur Verfügung. Zudem können überlappende geschützte Bereiche aufsteigend priorisiert werden, wobei Sieben die höchste und Null die niedrigste Priorität darstellt.

Außerdem können, wie bereits thematisiert, Zugriffsberechtigungen festgelegt werden. Alle Zugriffe auf die bestimmten Speicherbereiche werden durch Zugriffserlaubnisbits des Bereichszugriffssteuerregisters geregelt. Verletzungen dieser Regeln und nicht übereinstimmende MPUs rufen den MemManage-Fehlerbehandler mit programmierbarer Priorität auf [Ren18a]. Hierbei steht „Mem“ für Memory und das Memory Management Fault Status Register zeigt die Ursache von Speicherzugriffsfehlern an [ARM10a].

Bus Master MPU Die Bus Master MPU bietet Speicherschutzfunktionen für jeden Bus Master mit Ausnamhe der CPU und liegt in drei Gruppen A, B und C vor. Diese kontrollieren jeweils unterschiedliche dedizierte Bereiche des Speichers und arbeiten parallel. Eine Übersicht wird im Anhang in der Abbildung A.2 ersichtlich.

Die Bus Master MPU überwacht die Adressen im gesamten Adressraum (von 0000 0000h bis FFFF FFFFh), auf die die Bus Master zugreifen. Die Zugriffskontrolle, bestehend aus Lese- und Schreibberechtigungen, kann unabhängig für bis zu 32 Regionen konfiguriert werden. Hierbei umfassen die Regionen sowohl geschützte und zulässige Bereiche, welche sich überschneiden können als auch solche mit zwei überlappenden zulässigen Bereichen.

Dabei überwacht die Bus Master MPU den Zugriff auf jede Region basierend auf deren Einstellungen. Wenn ein Zugriff auf eine geschützte Region erkannt wird, erzeugt sie ein Reset oder einen nicht maskierbaren Interrupt. Nach dem Zurücksetzen setzt die Bus Master MPU die Berechtigung für alle Bereiche. Sollen alle Regionen geschützt werden, setzt man das Enable-Bit des Kontrollregisters auf Eins.

Bus Slave MPU Die Bus Slave MPU bietet Speicherschutzfunktionen für jeden Bus Slave. Sie überwacht den Zugriff auf die verschiedenen Speicherbereiche wie Flash oder SRAM, auf die von vier Busmastern aus zugegriffen werden kann: Zum einen von der CPU und zum anderen von den Bus Master MPU-Gruppen A, B und C. Die Bus Slave MPU hat ein separates Schutzregister für jeden der vier Bus Master mit unabhängiger Zugriffsschutzsteuerung, bestehend aus Lese- und Schreibberechtigung. Wenn ein Zugriff auf einen geschützten Bereich erkannt wird, erzeugt die MPU einen Reset oder einen nicht maskierbaren Interrupt und kann sowohl die Adresse, den Status als auch den Zugriffsstatus des Fehlers speichern.
Hierbei erfolgt die Überwachung der Bus Slave MPU und deren Zugriff ebenfalls basierend auf den Zugriffskontrolleinstellungen der jeweiligen geschützten Region. Wenn ein Zugriff auf die geschützte Region festgestellt wird, erzeugt die Bus Slave MPU einen Speicherschutzfehler [Ren18a].

4.3.3 Schutz des Flash-Speichers durch ID-Authentifizierung

Zur Programmierung des On-Chip-Flash-Speichers des S7G2 Mikrocontrollers kann sowohl eine serielle Schnittstelle (serielle Programmierung) als auch die JTAG/SWD-Schnittstelle (On-Chip-Debug-Modus) verwendet werden. Außerdem ist es möglich während des normalen Betriebsmodus einen dedizierten Flash-Speicherprogrammierer zu nutzen.
Zusätzlich sind Sicherheitsfunktionen eingebaut, um zu verhindern, dass Daten und Benutzerprogramme, welche sich im Chip-internen Flash-Speicher befinden, überschrieben werden. Zudem dienen sie dem Schutz vor eventuellen Manipulationen durch Dritte.

Um aufzuführen, wie der Flash-Speicher durch die ID-Authentifizierung geschützt werden kann, werden vorerst drei andere Begriffe, die verschiedenen Programmiermodi des Flash-Speichers, erklärt.

Selbstprogrammierung (Normaler Betriebsmodus) Dieser interne Modus ist vergleichbar mit einer Updatefunktion, welche innerhalb eines bestimmten Zeitraums nach Updates sucht. Er ist zunächst inaktiv und wird aktiviert sobald eine Aktualisierung gefunden wurde, welche er anschließend ausführt.
Solch ein Benutzerprogramm kann sowohl den Code-Flash-Speicher als auch den Daten-Flash-Speicher programmieren. Allerdings lediglich jene Code-Flash-

Bereiche, die durch dieses Programm im betreffenden Moment nicht genutzt werden. Die Fähigkeit, im Hintergrund zu agieren, ermöglicht es, Anweisungen entgegenzunehmen oder anderweitig Daten aus dem Code-Flash-Speicher zu lesen, während der Daten-Flash-Speicher programmiert wird.

Serielle Programmierung (Serieller Programmiermodus) Dieser Modus kann verwendet werden, um den Flash-Speicher durch externen Zugriff über eine serielle Schnittstelle zu programmieren. Voraussetzung dafür ist, dass das Gerät zuvor auf dem Zielsystem installiert wurde. Nutzt man zusätzlich eine Programmieradapterkarte, so ermöglicht diese die Veränderung des Flash-Speichers ohne eine vorherige Installation.

JTAG- oder SWD-Programmierung (On-Chip Debug Modus) Die JTAG- oder SWD-Programmierung findet unter den gleichen Voraussetzungen statt wie auch der serielle Programmiermodus, allerdings wird hierfür keine serielle Schnittstelle, sondern JTAG, SWD oder ein On-Chip-Debugger verwendet, um den Flash-Speicher zu programmieren.

Der On-Chip-Flash-Speicher unterstützt die ID-Code-Sicherheitsfunktion. Diese Authentifizierung durch ID-Codes wird bei serieller und bei JTAG- oder SWD-Programmierung verwendet.
Die ID-Code-Schutzfunktion verhindert die Programmierung und das On-Chip-Debugging. Wenn der ID-Code-Schutz aktiviert ist, validiert das Gerät den vom Host gesendeten ID-Code, indem er diesen mit dem im Flash-Speicher abgelegten ID-Code vergleicht. Sowohl die serielle Programmierung als auch das On-Chip-Debuggen werden nur dann erlaubt, wenn diese beiden Codes übereinstimmen.
Der ID-Code im Flash-Speicher des Boards besteht aus vier 32-Bit-Wörtern. Die ID-Code-Bits 126 und 127 bestimmen, ob der Schutz aktiviert ist und die Authentifizierungsmethode für den Host durchgeführt wird.

Ist das Ergebnis dieser ID-Authentifizierung positiv, kann es weiterführend verwendet werden, um die Verbindung des Controllers für die serielle Programmierung aufzubauen, zu steuern und zu kontrollieren. Dies betrifft jegliche Sicherheitseinstellungen und Lösch-, Programmier- und Lesevorgänge.
Während der seriellen Programmierung und dem On-Chip Debug-Modus gibt es verschiedene Sicherheitsfeatures: Wenn die ID-Codes nicht übereinstimmen, werden jegliche Zugriffsbefehle, beispielsweise jene, die Datenblöcke löschen, Programmierbefehle oder Lesebefehle, nicht unterstützt. Außerdem sind Konfigurationsbefehle, welche den Schutz und die Sicherheit des Controllers verändern würden, ebenfalls nicht erlaubt.

Die Voraussetzung dafür, dass die genannten Befehle funktionieren und unterstützt werden ist demnach, dass die ID-Codes übereinstimmen. Aufgrund dieser Methode können Daten, welche sich auf dem Chip befinden, während der seriellen Programmierung und dem Debuggen vor nicht authentifiziertem Zugriff geschützt werden.
Allerdings werden genau diese besagten Schutzmechanismen während des Selbstprogrammiermodus nicht unterstützt. Dafür wird die Interrupt-Verarbeitung während der Selbstprogrammierung unterstützt, so dass die Programmierung während der Verarbeitung externer Kommunikation und anderer Funktionen fortgesetzt werden kann [Ren18a].

4.3.4 Cyclic Redundancy Check (CRC) Calculator

Die zyklische Redundanzprüfung ermittelt einen Prüfwert, welcher Fehler bei der Datenübertragung oder -speicherung erkennbar macht. Anhand dessen lässt sich die Integrität der Daten prüfen. Zudem kann die Signatur der Software während der Laufzeit verifiziert werden.
Der Cyclic Redundancy Check-Rechner generiert CRC-Codes. Die Bit-Reihenfolge der CRC-Berechnungsergebnisse kann zwischen LSB und MSB umgeschaltet werden – je nachdem, ob das niederwertige oder das höherwertige Byte des Codes zuerst übertragen wird.
Zusätzlich ermöglicht die sogenannte Snoop-Funktion die Überwachung von Lese- und Schreibvorgängen an bestimmten Adressen, indem der Vergleich der CRC-Ergebniscodes automatisch durchgeführt wird. Diese Funktion ist in Anwendungen nützlich, in denen CRC-Code bei bestimmten Ereignissen automatisch generiert werden muss, beispielsweise beim Überwachen von Schreibvorgängen im seriellen Übertragungspuffer oder beim Lesen aus dem seriellen Empfangspuffer. Wenn das CRCSWR-Bit (Snoop-On-Write/Read Switch) auf Null gesetzt ist, wird die Snoop-Operation zum Lesen einer spezifischen Registeradresse aktiviert. Ist es auf Eins gesetzt, wird die Operation verwendet, um in eine bestimmte Registeradresse zu schreiben [Ren18a].

5 Diskussion

Um den Renesas Synergy S7G2 Mikrocontroller und dessen Funktionen einschätzen zu können, wird dieser nachfolgend mit dem ATXMEGA-A3BU-Xplained der Firma Atmel verglichen.
Vorerst soll im Folgenden der ATXMEGA-A3BU-Xplained Mikrocontroller mit seinen Komponenten und Funktionen näher beschrieben werden.

5.1 Der Atmel ATXMEGA-A3BU-Xplained Mikrocontroller

Merkmale und Funktionen

Der Atmel ATXMEGA-A3BU-Xplained Mikrocontroller umfasst folgende Merkmale und Funktionen:

- **Display:**
  - 1,3 Zoll FSTN-LCD-Display (128x32 Pixel) [Atm12]
- **Drahtgebundene Konnektivität:**
  - USB-Geräteschnittstelle
  - sechs USART-Schnittstellen
  - zwei Inter-Integrated Circuit- bzw. Two-Wire-Interfaces und zwei serielle Peripherieschnittstellen
- **Speicher:**
  - 256 kB Code Flash
  - 4 kB EEPROM
  - 16 kB interner SRAM
  - 8 kB Boot-Code-Bereich

- **Sicherheit:**
  - AES und DES Verschlüsselung
  - Programm- und Debug-Schnittstelle (PDI)
  - Batterie-Backup-System
  - Hardwarediagnose, Speicherschutz und Watchdog Timeout Debugging mit separatem On-Chip Ultra Low Power Oszillator
  - Brownout-Erkennung[1] [Atm14]

Im Folgenden werden verschiedene Bausteine und deren Funktionen näher erklärt. Dabei liegt das Hauptaugenmerk sowohl auf denen, die sicherheitsrelevant sind bzw. sein könnten als auch auf denen, die unter Punkt 2.3.1 bereits thematisiert wurden.

CPU Der besagte Atmel Mikrocontroller besitzt einen Mikroprozessor, der auf der AVR-erweiterten RISC-Architektur[2] basiert und eine Taktrate von bis zu 32 Mhz erreicht. Zudem gibt es zur Behandlung von möglichen Ausnahmefällen und Unterbrechungen einen programmierbaren Multilevel-Interrupt-Controller. Der Mikrocontroller der Firma Atmel enthält außerdem einen Direct Memory Access Controller. Dieser Zugriffscontroller ermöglicht es, Daten zwischen Speichern und Peripheriegeräten zu übertragen und übernimmt so Aufgaben der CPU. Dadurch wird eine hohe Datenübertragungsrate bei minimalem CPU-Eingriff ermöglicht und gleichzeitig der Prozessor entlastet. Hierbei ermöglichen vier Kanäle bis zu vier unabhängige und parallele Übertragungen.

Speicher Auf dem Chip des ATXMEGA-A3BU-Xplained befindet sich ein EEPROM, welcher auch unter E^{2}PROM bekannt ist, und für Electrically Erasable Programmable Read Only Memory steht. Er stellt einen Speicher dar, welcher vom Prozessor lediglich gelesen, jedoch nicht beschrieben werden kann und zählt zu den nicht flüchtigen Speichern [Bäh10]. Neben 256 kB Code Flash verfügt der ATXMEGA-A3BU-Xplained über einen 8 kB Boot-Code-Bereich. Zudem hat der Mikrocontroller einen 16 kB großen internen SRAM, welcher zu den Bausteinen mit einem flüchtigen Speicher gehört [Atm14].

[1] BOD = Brownout Detection, wird genutzt, um den Pegel der Versorgungsspannung während des Betriebs zu überwachen. Dies geschieht durch einen automatischen Vergleich dessen mit einem programmierbarem Pegel [Atm14].

[2] AVR ist eine Mikrocontroller-Familie des Herstellers Mikrochip. Da innerhalb der RISC-Architektur nur reduzierte Befehlssätze verwendet werden, können die meisten Register-Befehle vergleichsweise schnell – während eines Systemtakts – ausgeführt werden [BM08].

Energieversorgung Die Stromversorgung des ATXMEGA-A3BU-Xplained erfolgt über den USB-Anschluss, welcher sich oberhalb des Displays befindet, oder über den Pin 10 am Jumper J3. Vorzugsweise wird der USB-Anschluss genutzt, da zusätzliche Erweiterungskarten so am Jumper J3 angeschlossen werden können. Zudem verfügt der ATXMEGA-A3BU-Xplained über ein Batterie-Backup-System, durch welches automatisch von der Haupt- zur Notstromversorgung umgeschaltet wird, wenn der Hauptakku oder die externe Stromquelle nicht verfügbar sind. Das System unterstützt die Nutzung der Backup-Batterie, sobald der dedizierte VBAT-Pin aktiviert ist, welcher sich neben der Batterie befindet.

Peripheriegeräte Der ATXMEGA-A3BU-Xplained beinhaltet neben einem USB-Anschluss und zwei Two-Wire-Interfaces ebenso zwei serielle Peripherieschnittstellen. Dabei sind die Zweidraht-Schnittstellen vergleichbar mit den bereits thematisierten I^2Cs, welche serielle Busse zur Datenübertragung darstellen.
Weiterführend besitzt der ATXMEGA-A3BU-Xplained zum Senden und Empfangen von Daten sechs USART-Schnittstellen. Universal Synchronous/ Asynchronous Receiver Transmitter sind Bausteine, welche sowohl synchron als auch asynchron Daten übertragen können.

Schutz und Sicherheit Der ATXMEGA-A3BU-Xplained verfügt für die Programmierung und die Fehlersuche sowohl über eine Programm- und Debug-Schnittstelle als auch über ein JTAG-Interface. Der Mikrocontroller der Firma Atmel beinhaltet außerdem sogenannte Lock-Bits, welche genutzt werden können, um bestimmte Bereiche vor Lese- und Schreibzugriffen zu schützen. Weiterführend enthält der ATXMEGA-A3BU-Xplained zur Gewährleistung einer permanenten Stromversorgung, trotz beispielsweise eines Stromausfalls, das oben bereits genannte Batterie-Backup-System [Atm14].

Die Abbildung 5.1 zeigt die Position aller relevanten Komponenten des ATXMEGA-A3BU-Xplained Mikrocontrollers, die im vorherigen Abschnitt genannt und näher erläutert wurden.

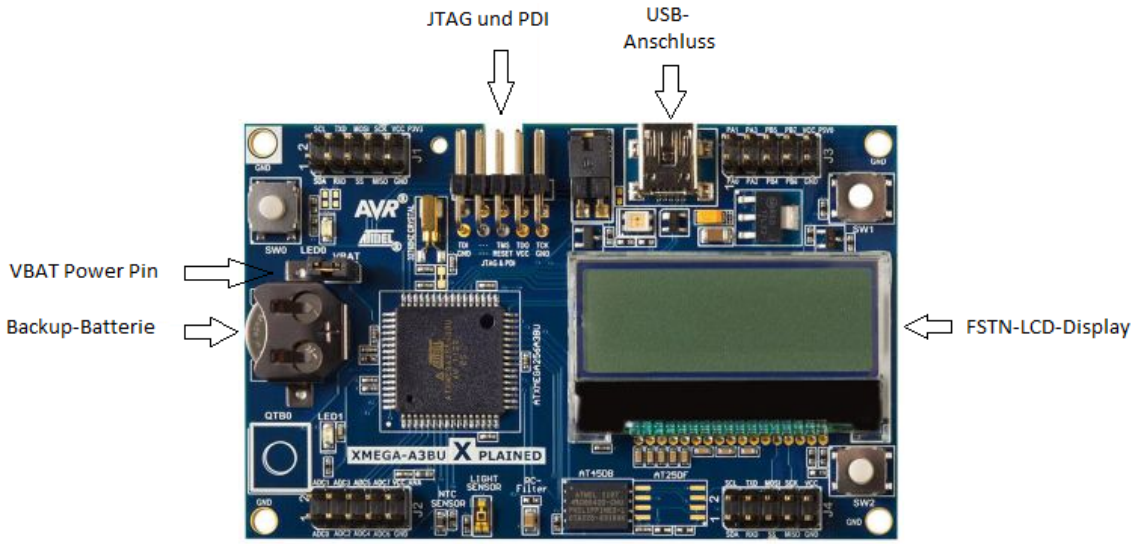

Abbildung 5.1: Layout des ATXMEGA-A3BU-Xplained-Boards [eigene Arbeit, modifiziert nach [Eas11]]

5.2 Theoretischer Vergleich des Renesas Synergy S7G2s mit dem Atmel ATXMEGA-A3BU-Xplained

Im Folgenden soll der ATXMEGA-A3BU-Xplained Mikrocontroller der Firma Atmel vergleichend hinzugezogen werden, um so die Sicherheit des Renesas Synergy S7G2s und dessen Schutzfunktionen zu diskutieren. Dabei wird Bezug genommen auf die unter Punkt 3.2.4 und 4.3 genannten Merkmale des Mikrocontrollers.

5.2.1 Schreibschutz

Eine der wichtigsten Schutzfunktionen, die der Renesas Synergy S7G2 bietet, ist der Schreibschutz für bestimmte Register, um diese vor unzulässigem Schreibzugriff zu schützen. Dies ist mit Hilfe des Protect Registers möglich, indem man das Protect-Bit des jeweiligen Registers aktiviert [Ren18a].

Eine ähnliche Funktion bietet auch der Atmel ATXMEGA-A3BU-Xplained. Während sich beim Renesas Synergy S7G2 die Schreibschutzfunktionen auf drei konkrete

verschiedene Regionen — die Clock Generation Circuit-, die Energiesparmodi- und Batterie-Backup- und außerdem die Niederspannungserkennungsregister — beziehen, beinhaltet der Controller von Atmel sogenannte Verriegelungsbits (engl. Lock Bits), welche auf den Speicher allgemein angewendet werden können [KK14].

Der Flash-Speicher des Atmel Mikrocontrollers ist in zwei Abschnitte geteilt — den Anwendungs- und den Bootloader-Abschnitt. Für beide Regionen sind dedizierte Verriegelungsbits vorhanden, welche unterschiedliche Schutzebenen bieten. Der Befehl, welcher zum Schreiben in den Flash-Speicher von der Anwendungssoftware verwendet wird, funktioniert nur, wenn er vom Bootloader-Abschnitt aus ausgeführt wird. Hierbei kann Zugriff auf den gesamten Flash-Speicher inklusive des Bootloader-Abschnitts erlangt werden [Atm14]. Genau diese Tatsache macht es besonders wichtig, uneingeschränkte Zugriffe auf die Anwendungen und die gespeicherten Daten zu verhindern.

Dafür sind zwei verschiedene Arten von Verriegelungsbits vorhanden: Die Boot-Lock-Bits und die „normalen“ Lock Bits. Sowohl die Schutzstufe des Bootloader-Abschnitts als auch die des Anwendungsabschnitts kann durch die Boot-Look-Bits ausgewählt werden. Dadurch kann der Anwendungsbereich vor Zugriffen durch den Bootloader geschützt werden. Zudem kann der Flash-Speicher und der EEPROM durch die Lock Bits gegenüber Lese- und Schreibzugriffen geschützt werden [Wal09].
Außerdem enthält der Anwendungsabschnitt eine Tabelle mit separaten Sperreinstellungen. Hierbei sperrt ein Sicherungs- oder Lock-Bit, welches den Wert Eins hat, die Berechtigung, während ein Bit mit dem Wert Null Zugriffe zulässt [Atm14].

Zusammenfassend für diesen Abschnitt lässt sich demnach sagen, dass sowohl der Renesas Synergy S7G2 als auch der Atmel ATXMEGA-A3BU-Xplained über Protection- bzw. Lock-Bits verfügen, um den Schreibzugriff für bestimmte Register oder Abschnitte zu deaktivieren oder zu aktivieren. Bei beiden Controllern sind diese Einstellungen nicht voreingestellt, sondern können nachträglich aktiviert und konfiguriert werden.

5.2.2 Memory Protection Unit

Weiterführend weist der Atmel ATXMEGA-A3BU-Xplained, anders als der S7G2, keine Memory Protection Unit auf.
Die MPU des Renesas Controllers stellt einen beträchtlichen Speicherschutz zur Verfügung, indem sie Transaktionen, Datenzugriffe und Befehlsabrufe überwacht und gleichzeitig Interrupts auslöst, sobald eine Zugriffsverletzung erkannt wird. Zudem können mit Hilfe der MPU Speicherbereiche definiert und Zugriffsberechtigungen

festgelegt werden. Verhindert werden soll dadurch, dass beispielsweise Fehler oder Schadsoftware einen Prozess oder gar das Betriebssystem beeinflussen, indem sie auf einen bestimmten Speicherbereich zugreifen, der ihnen nicht zugewiesen wurde. Hierbei verfügt der S7G2 sowohl über drei MPUs als auch über eine CPU Stack Pointer Monitor-Funktion, welche den Stack Pointer überwacht. Er ist in der Lage, Stack Over- und Underflows zu detektieren und schützt so den Mikrocontroller vor Angriffen [Ren18a].

Da der Atmel ATXMEGA-A3BU-Xplained diese entscheidende Speicherschutzfunktion nicht aufweist, stellt es einen Vorteil für die Sicherheit des Renesas Mikrocontrollers dar. Besonders der Schutz vor Malware-Angriffen ist wichtig, da die Zahl solcher immer mehr zunimmt und allein von 2016 auf 2017 um 18,4 Prozent gestiegen ist. So wurden im Jahr 2017 nach [Pat18] weltweit 9,32 Milliarden Angriffe durch Schadsoftware erkannt. Allerdings muss dabei berücksichtigt werden, dass es zudem eine gewisse Dunkelziffer gibt, welche hier nicht aufgeführt wird.

5.2.3 Cyclic Redundancy Check Calculator

Der Renesas Synergy S7G2 verfügt, wie auch der Atmel ATXMEGA-A3BU-Xplained, über einen Generator zur zyklischen Redundanzprüfung. Dieser detektiert auftretende Fehler innerhalb der Datenübertragung oder -speicherung und bestimmt somit die Korrektheit der Daten.

Während beim S7G2 eine zusätzliche Snoop-Funktion vorhanden ist, mit der die Überwachung von Lese- und Schreibvorgängen durch einen automatischen Vergleich der CRC-Ergebnisse gesondert durchgeführt wird, verfügt der Atmel ATXMEGA-A3BU-Xplained lediglich über die Standard-CRC-Funktion [Ren18a]. Hierbei erzeugt der CRC-Generator aus einem eingegebenen Datenstrom eine Prüfsumme, welche an die Daten angehängt wird. Werden genau diese Daten weiterführend empfangen oder gelesen, dann wird die Berechnung der Prüfsumme wiederholt und das daraus resultierende Ergebnis mit dem vorherigen verglichen. Wird dabei keine Übereinstimmung festgestellt, so enthält der zugrundeliegende Datenstrom einen Fehler. Das erkennt der Calculator und kann eine jeweilige Korrekturmaßnahme ergreifen, indem er beispielsweise die Daten erneut sendet [Atm14].

Resümierend in Bezug auf diesen Absatz ist festzustellen, dass sowohl der Renesas Synergy S7G2 als auch der Atmel ATXMEGA-A3BU-Xplained über einen Cyclic Redundancy Check Calculator verfügen, um Fehler zu erkennen und eine Datenkorrektheit zu gewährleisten. Allerdings bietet der S7G2 eine zusätzliche Schutzfunktion,

um darüber hinaus Vorgänge an bestimmten Adressen überwachen zu können. Diese ergänzende Snoop-Funktion ist im Atmel Controller nicht integriert.

5.2.4 JTAG

Eine weitere Gemeinsamkeit, die bei beiden betrachteten Mikrocontrollern vorliegt, ist das Vorhandensein einer JTAG-Schnittstelle, um Hardware zu programmieren und zu debuggen, d.h. Programme zu testen und so Fehler detektieren zu können. Hierbei verfügen sowohl der Renesas S7G2 als auch der Atmel ATXMEGA-A3BU-Xplained über verschiedene Mechanismen, um den JTAG-Zugriff zu aktivieren bzw. zu deaktivieren.
Um JTAG anzuwenden, bietet der Renesas-Controller zwei Möglichkeiten: Einerseits den indirekten Zugang durch einen bereits integrierten J-Link Debugger, welcher über den USB-Anschluss J19 zugänglich ist, andererseits den direkten Zugriff über den Anschluss J18. Während der indirekte Anschluss ohne weitere Vorkehrungen nutzbar ist, muss der direkte Zugang erst aktiviert werden. Dies ist möglich, indem vier Widerstände entfernt werden. Beachtet werden muss außerdem, dass danach der indirekte Zugriff nicht mehr verwendet werden kann, außer die jeweiligen Widerstände werden wieder passgenau auf den Controller gelötet [Ren15].

Auch der ATXMEGA-A3BU-Xplained verfügt über gewisse Einschränkungen, die den JTAG-Zugriff schützen sollen. Ein Unterschied besteht allerdings darin, dass hier ein externes Programmier- und Debugging-Tool an den „JTAG und PDI“-Anschluss angeschlossen werden muss. Zur Verfügung stehen dabei mehrere Tools, von denen im Folgenden lediglich drei betrachtet werden: *Atmel AVR ONE!*, *Atmel JTAGICE 3* und *Atmel AVR JTAGICE mkII*. All diese Enwicklungswerkzeuge werden genutzt, um Atmel 8- und 32-Bit-AVR-Geräte, welche mit der On-Chip-Debug-Funktion ausgestattet sind, zu programmieren.
Allerdings müssen hierbei verschiedene Voraussetzungen beachtet werden. So muss man beispielsweise, um JTAG mit Hilfe von *JTAGICE mkll* zu verwenden, ein sogenanntes spezielles Suqid cable („Tintenfisch-Kabel“) verwenden. Außerdem wird ein Abstandsadapter benötigt, um JTAG mit *Atmel AVR ONE!* zu nutzen. Neben JTAG kann am selben Anschluss auch PDI (Periphäre Debugging-Schnittstelle) verwendet werden, was aber nur funktioniert, wenn JTAG auf dem Gerät deaktiviert wurde. Der Grund dafür ist, dass, wenn JTAG aktiviert ist, ein Pull-Up Widerstand[3] ermöglicht

[3]Ein Pull-Up („hoch ziehen“) Widerstand wird genutzt, um an einem Eingang den Wert High zu erreichen. Ein Eingang ist im Normalzustand hochohmig, d.h. zwischen High und Low. Aufgrund dessen, dass Schaltungen durch Störsignale behindert werden können, kommt es zu Über- oder Unterschreitungen eines bestimmten Wertes, wodurch wiederum Fehler auftreten [Bra09].

wird, was wiederum die PDI-Initialisierungssequenz stört.
Dennoch ist es möglich, den Pin zu verwenden. Dafür muss allerdings das Band J203, welches das TDO- und das PDI-DATA-Signal verbindet und sich auf der Rückseite der Platine befindet, durchtrennt werden. Dadurch wird jedoch die JTAG-Schnittstelle so lange deaktiviert, bis die Verbindung wiederhergestellt ist, beispielsweise durch das Anlöten einer Brücke an jenes durchtrennte Band [Atm12].

Zusammenfassend für diesen Abschnitt lässt sich demnach sagen, dass sowohl der Renesas Synergy S7G2 als auch der Atmel ATXMEGA-A3BU-Xplained über Debugging-Schnittstellen, insbesondere über JTAG, verfügen, um Programme testen und Fehler innerhalb derer detektieren zu können. Hierbei sind bei beiden Controllern einige Voraussetzungen bzw. Einschränkungen gegeben. Während beim S7G2 für den direkten JTAG-Zugang Widerstände abgelötet werden müssen, muss beim ATXMEGA-A3BU-Xplained ein Verbindungsband durchtrennt werden, um PDI zu nutzen. Der Unterschied besteht allerdings darin, dass beim Atmel Controller der JTAG-Zugang genutzt werden kann, aber der PDI-Zugriff erst ermöglicht werden muss, während beim Renesas-Controller nur ein bestimmter JTAG-Zugriff – der direkte – durch Veränderungen am Board aktiviert werden muss.
Nutzbar ohne zusätzliche Vorkehrungen ist die JTAG-Schnittstelle aber generell standardmäßig bei beiden Controllern, zumindest die Nutzung des indirekten Zugangs innerhalb des Renesas S7G2s.

5.3 Zusammenfassung

Der Renesas Synergy S7G2 Mikrocontroller bietet neben den Standard-Hardwarekomponenten und -Schnittstellen eines Mikrocontrollers verschiedene Erweiterungen und Extrafunktionen, wie beispielsweise zwei Pmod-Anschlüsse, BLE oder ein Farbdisplay.
Zudem kann der S7G2 durch eine Vielzahl an Software ergänzt werden. Diese soll einerseits mehr Sicherheit bieten und andererseits die Nutzungs- und Kommunikationsfunktionen erweitern. Doch auch hier gilt: Je mehr zusätzliche Konnektivität, wie z.B. durch Wireless Application Frameworks wie Wi-Fi, LTE Mobilfunk und Bluetooth Low Energy, desto mehr Angriffsmöglichkeiten.

Um einen gewissen Schutz zu bieten, ist der S7G2 Controller mit verschiedenen Sicherheitsmechanismen ausgestattet. Diese betreffen sowohl Hard- als auch Software. Die Schutzfunktion der Nutzung einer Memory Protection Unit und die Ergänzung dieser um einen zusätzlichen CPU Stack Pointer Monitor ist ausschlaggebend für die

Sicherheit. Durch beide Komponenten wird der Speicher vor eventuellen Angriffen, durch beispielsweise Schadsoftware oder durch entstehende Fehler, geschützt.
Weiteren Schutz bietet zudem der Registerschreibschutz, welcher für bestimmte Register aktiviert werden kann und unbefugten Schreibzugriff verhindert. Während der Datenübertragung oder -speicherung sorgt weiterhin eine zyklische Redundanzprüfung dafür, dass die Integrität der Daten gewahrt und sichergestellt wird. Dies wird ergänzt durch eine Snoop-Funktion, um Lese- und Schreibvorgänge an konkreten Adressen zu überwachen.

All diese Eigenschaften und Schutzmechanismen wurden mit den Funktionen und Merkmalen des Atmel ATXMEGA-A3BU-Xplained Mikrocontrollers verglichen. Anhand des erfolgten Vergleichs mit jenem Mikrocontroller lässt sich sagen, dass beide eine gewisse Grundsicherheit bieten, da sie sich in vielen Punkten ähneln. Beispielsweise besitzt der ATXMEGA-A3BU-Xplained auch einen CRC-Generator und Lock Bits, um Fehler zu detektieren und bestimmte Regionen im Speicher gesondert zu schützen. Allerdings verdeutlicht der Vergleich, dass der Renesas Synergy S7G2 mehr Schutzfunktionen bietet als der Atmel ATXMEGA-A3BU-Xplained.

5.4 Ausblick

Wie bereits thematisiert, gibt es sehr viele verschiedene Möglichkeiten, um in ein System einzudringen und es gibt ebenso eine Vielzahl unterschiedlicher Methoden, um die Systeme gegen Angriffe zu schützen und Sicherheitslücken zu beheben.
Im Rahmen dieser Bachelorarbeit wurden ausgewählte Funktionen des Renesas Synergy S7G2 Mikrocontrollers betrachtet. Hierbei lag das Hauptaugenmerk des praktischen Tests vor allem auf der Durchführung eines Boundary-Scans und der damit verbundenen JTAG-/SWD-Schnittstelle.
Theoretisch wurden ebenfalls nur einige ausgewählte Schutzmechanismen beleuchtet, so dass sowohl praktisch als auch theoretisch noch mehr Tests und Untersuchungen möglich sind. Gerade weil der S7G2, wie in dieser Arbeit dargestellt, so viele verschiedene Funktionen, Schnittstellen und Erweiterungen bietet, sind weitere Tests in Bezug auf die Sicherheit derer zu erwarten.

Zudem ist anzubringen, dass im Rahmen dieser Bachelorarbeit lediglich der Vergleich mit einem anderen Mikrocontroller – dem Atmel ATXMEGA-A3BU-Xplained – stattgefunden hat. Demnach sind weiterführend Vergleiche mit weiteren MCUs möglich. Diese Vergleiche können ebenfalls innerhalb der ausgewählten thematisierten Schutzmechanismen erweitert werden.

Anhang A: Abbildungen

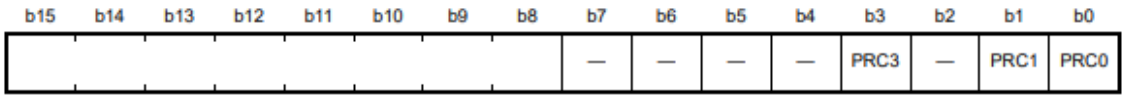

Abbildung A.1: Aufbau des PRC-Registers [eigene Arbeit, modifiziert nach [Ren18a]]

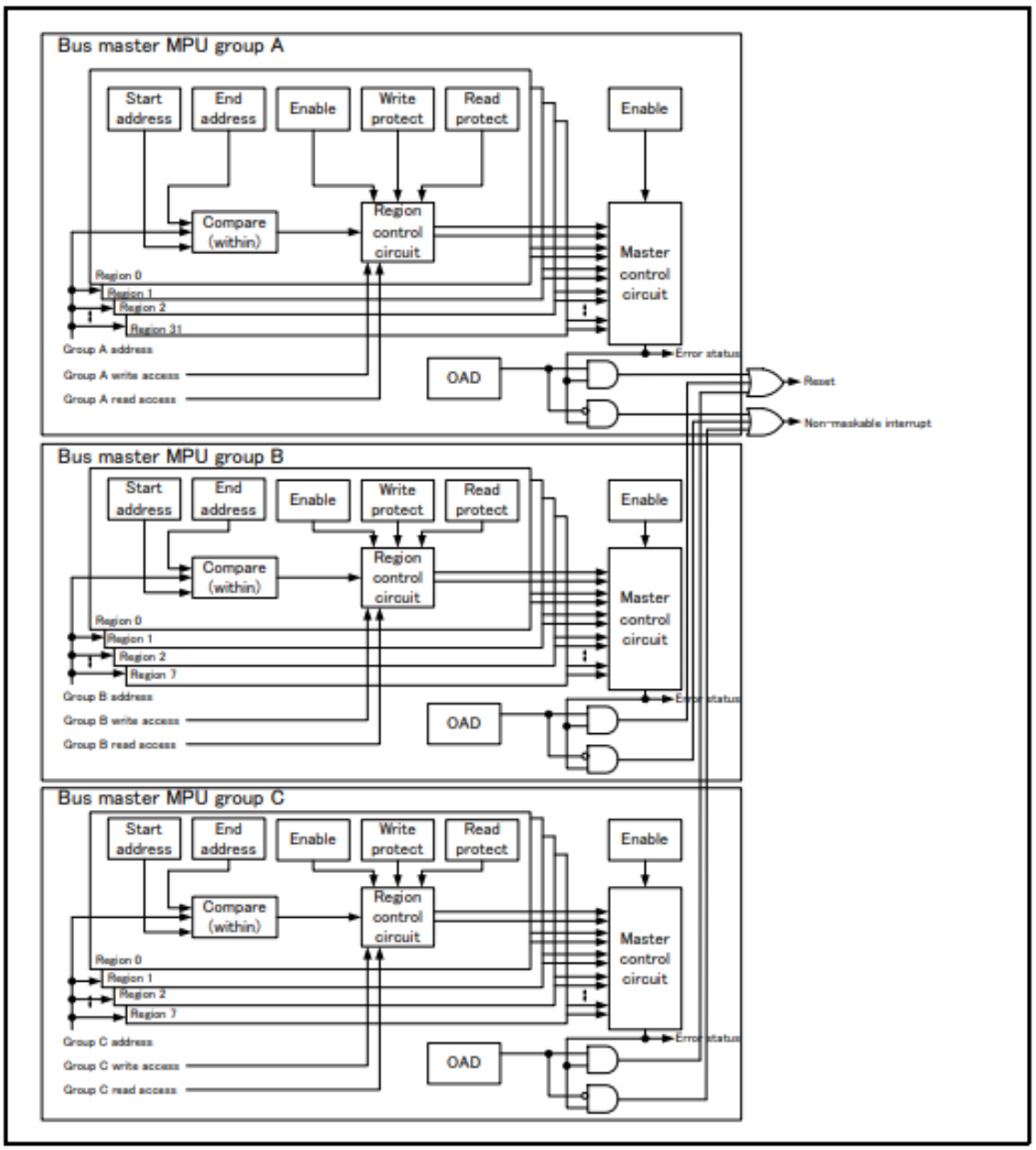

Abbildung A.2: Bus Master MPU Gruppen A,B, und C [Ren18a]

Anhang B: Tabellen

Jumper	Funktion
J1	**Boot-Auswahl:** Ist J1 in Position 1-2 (Standard), bootet die MCU im normalen Modus (von ihrem ROM). Ist J1 auf Position 2-3 eingestellt, bootet die MCU im USB-Programmiermodus und der MCU-Flash kann über den USB-Port programmiert werden.
J2	**Reset:** Ist J2 gesetzt, wird die MCU zurückgesetzt. Ist J2 offen, wird der Reset freigegeben.
J8	**RS-232-Sendebereitschaft/RS-485-Empfangsfreigabe:** Ist J8 in Position 1-2 (Standard), ist das Sendebereitschaftssignal des RS-232-Treibers (das anzeigt, dass die Übertragungsspannungen angemessen sind) mit einem GPIO-Signal der MCU verbunden. Ist J8 in Position 2-3, ist die Empfangsfreigabe des (externen) RS-485-Treibers mit demselben GPIO-Signal verbunden.
J9	**RS-232/485 Modusauswahl:** Ist J9 in den Positionen 1-3 und 2-4 (Standard), wird der SCI3-Port der MCU mit dem RS-232-Treiber verbunden (liegt am J7-Port frei). Ist J9 in der Position 3-5 und 4-6, ist der SCI3-Port der MCU mit dem RS-485-Port (J10) für einen externen Treiber verbunden.
J13	**PMODA Ausgangsauswahl:** Ist J13 auf „5V“ eingestellt, werden 5V im PMODA-Anschluss bereitgestellt. Ist J13 auf „3V3“ eingestellt, werden 3,3 V in der PMODA-Schnittstelle bereitgestellt.
J15	**PMODB Ausgangsauswahl:** Ist J15 auf „5V“ eingestellt, sind im PMODB-Anschluss 5V vorgesehen. Ist J15 auf „3V3“ eingestellt, werden 3,3 V in der PMODB-Schnittstelle bereitgestellt.
J31	**Leistungsmessung:** Wenn J31 installiert ist, läuft der MCU-Strom durch den Jumper. Wenn J31 entfernt wird, fließt der Stromverbrauch der MCU durch ein Amperemeter, das an die Pins 1 und 2 von J31 angeschlossen ist.

Tabelle B.1: Konfigurationsoptionen mittels Jumper [modifiziert nach [Ren15]]

Option	**J8**	**J9**
RS-232 am J7	1-2	1-3 und 2-4
RS-485 auf J10 (externer Konverter)	2-3	3-5 und 4-6

Tabelle B.2: Konfiguration des RS-232-Transceivers [modifiziert nach [Ren15]]

Literaturverzeichnis

[ARM10a] ARM Limited, Hrsg. *Cortex-M4 Devices Generic User Guide*. 16. Dez. 2010. URL: http://infocenter.arm.com/help/topic/com.arm.doc.dui0553a/DUI0553A_cortex_m4_dgug.pdf.

[ARM18a] ARM Limited, Hrsg. *Downloads - GNU Arm Embedded Toolchain*. 27. Juni 2018. URL: https://developer.arm.com/open-source/gnu-toolchain/gnu-rm/downloads (besucht am 12. 07. 2018).

[ARM10b] ARM Limited, Hrsg. *ARMv7-M Architecture Reference Manual*. 2006-2010. URL: https://web.eecs.umich.edu/~prabal/teaching/eecs373-f10/readings/ARMv7-M_ARM.pdf.

[ARM18b] ARM Limited, Hrsg. *2-Pin Debug Port*. 1995-2018. URL: https://developer.arm.com/products/system-ip/coresight-debug-and-trace/coresight-architecture/serial-wire-debug (besucht am 26. 07. 2018).

[Atm12] Atmel Corporation, Hrsg. *Atmel AVR1923: XMEGA-A3BU Xplained Hardware User Guide*. 2012. URL: http://ww1.microchip.com/downloads/en/DeviceDoc/doc8394.pdf (besucht am 30. 09. 2018).

[Atm14] Atmel Corporation, Hrsg. *Atmel XMEGA A3BU Mikrocontroller Datasheet*. 2014. URL: http://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-8362-8-and-16bit-AVR-microcontroller-ATxmega256A3BU_datasheet.pdf (besucht am 30. 09. 2018).

[Ay18] Cengiz Ay. *HAL: Hardware Abstraction Layer*. 2018. URL: https://www.edv-lehrgang.de/hardware-abstraction-layer/ (besucht am 17. 05. 2018).

[Bäh10] Helmut Bähring. *Anwendungsorientierte Mikroprozessoren. Mikrocontroller und Digitale Signalprozessoren*. Bd. 4. Auflage. Springer-Verlag Berlin Heidelberg, 11. Juni 2010.

[BM08] Bernd Becker und Paul Molitor. *Technische Informatik. Eine einführende Darstellung*. Oldenbourg Wissenschaftsverlag GmbH München, 17. März 2008.

[BH09] Juliane Benra und Wolgang A. Halang. *Software-Entwicklung fur Echtzeitsysteme*. 1. Auflage. Springer-Verlag Berlin Heidelberg, 31. Aug. 2009.

[Bra09] Frank Brall. *Pullup Pulldown Widerstand*. Hrsg. von www.rn-wissen.de. 2. Nov. 2009. URL: https://rn-wissen.de/wiki/index.php/Pullup_Pulldown_Widerstand (besucht am 01. 09. 2018).

[BU10] Uwe Brinkschulte und Theo Ungerer. *Mikrocontroller und Mikroprozessoren*. 3. Auflage. Springer-Verlag Berlin Heidelberg, 5. Aug. 2010.

[Czi15] Horst Czichos. *Mechatronik. Grundlagen und Anwendungen technischer Systeme*. 3. Auflage. Springer Vieweg Wiesbaden, 27. Sep. 2015.

[Eas11] EasyElectronics.ru, Hrsg. *AVR Studio 5.0 und XMEGA256A3. Der Anfang.* 7. Nov. 2011. URL: http://we.easyelectronics.ru/AVR/avr-studio-50-i-xmega256a3-nachalo.html (besucht am 21. 08. 2018).

[Eic90] Herbert Eichele. *Multiprozessorsysteme. Eine Einführung in die Konzepte der Modernen Mikrocomputer- und Rechnertechnologie*. Vieweg +Teubner Verlag, 1. Apr. 1990.

[Exp18] Express Logic, Hrsg. *Express Logic - About Us*. 2018. URL: https://rtos.com/about/ (besucht am 17. 05. 2018).

[Fel11] Sven Feldkord. *Scheduling: Theorie und Praxis am Linux Kernel 2.6*. 1. Auflage. Grin Verlag, 4. Apr. 2011.

[FS12] Max Fischer und Rainer Seck. *Scheduling*. 15. Mai 2012. URL: https://pdfs.semanticscholar.org/presentation/6516/616a7c30c3ab5ab5122b76b7922152e001b8.pdf (besucht am 27. 05. 2018).

[FRR13] Stephan Fischer, Christoph Rensing und Utz Rödig. *Open Internet Security. Von den Grundlagen zu den Anwendungen*. 1. Auflage. Springer-Verlag Berlin Heidelberg, 4. Okt. 2013.

[Hag15] Jens Hagel. *Was ist ein Portscan?* 1. Mai 2015. URL: https://www.hagel-it.de/it-service/was-ist-ein-portscan.html (besucht am 26. 06. 2018).

[Hai07] Marc Haisenko. *GDB Cheatsheet*. 2007. URL: https://darkdust.net/files/GDB%20Cheat%20Sheet.pdf (besucht am 03. 09. 2018).

[ITW14] ITWissen, Hrsg. *Master-Slave-Betrieb*. 28. Jan. 2014. URL: https://www.itwissen.info/Master-Slave-Betrieb-master-slave-operation.html (besucht am 27. 07. 2018).

[ITW17] ITWissen, Hrsg. *Unterbrechungsanforderung*. 9. Feb. 2017. URL: https://www.itwissen.info/Unterbrechungsanforderung-interrupt-request-IRQ.html (besucht am 16. 05. 2018).

[JTA16] JTAG Technologies, Hrsg. *Über Boundary-Scan*. 2016. URL: https://www.jtag.com/de/content/about-boundary-scan (besucht am 06. 07. 2018).

[KK14] Markus Krauße und Rainer Konrad. *Drahtlose ZigBee-Netzwerke. Ein Kompendium*. Springer Vieweg, 21. Nov. 2014. URL: https://link.springer.com/content/pdf/10.1007%2F978-3-658-05821-0.pdf.

[Mar08] Peter Marwedel. *Eingebettete Systeme*. 1. Auflage. Springer-Verlag Berlin Heidelberg, 8. Aug. 2008.

[MT92] Colin M. Maunder und Rodham E. Tulloss. *IEEE Design and Test of Computers*. Bd. 9. Auflage. IEEE Computer Society Press, 1992.

[Nic18] Jean P. Nicolle. *How JTAG works*. 2018. URL: https://www.fpga4fun.com/SiteInformation.html (besucht am 26. 07. 2018).

[Nor18] Nordic Semiconductor, Hrsg. *QSPI — Quad serial peripheral interface*. 2018. URL: http://infocenter.nordicsemi.com/index.jsp?topic=%2Fcom.nordic.infocenter.nrf52840.ps%2Fqspi.html (besucht am 21. 05. 2018).

[NM05] J. Norhuzaimin und H. H. Maimun. "The design of high speed UART". In: IEEE, 20. Dez. 2005. URL: https://ieeexplore.ieee.org/document/1607831/.

[Oed17] Richard Oed. *Basics of the Renesas Synergy Platform*. Renesas Electronics Europe GmbH, 2017.

[Pat18] Ulrich Pathier. *Cyber-Angriffe entwickelten sich 2017 zum größten Geschäftsrisiko*. Hrsg. von IT Verlag für Informationstechnik GmbH. 6. März 2018. URL: https://www.it-daily.net/analysen/18097-cyber-angriffe-entwickelten-sich-2017-zum-groessten-geschaeftsrisiko (besucht am 22. 08. 2018).

[PM06] Jean Picard und Barry Jon Male. "Methods of inter-integrated circuit addressing and devices for performing the same". US7587539B2. 25. Apr. 2006. URL: https://patents.google.com/patent/US7587539B2/en.

[Ren15] Renesas Electronics Corporation, Hrsg. *Sk-S7G2 User's Manual*. 8. Okt. 2015. URL: https://www.renesas.com/us/en/doc/products/renesas-synergy/doc/r12um0004eu0100_synergy_sk_s7g2.pdf.

[Ren17] Renesas Electronics Corporation, Hrsg. *Getting Started Guide for Heartbeat Kit*. 29. Juni 2017. URL: https://synergygallery.renesas.com/media/products/88/210/en-US/r12an0063eu0100-synergy-sk-s7g2-heartbeat.pdf.

[Ren18a] Renesas Electronics Corporation, Hrsg. *S7G2 Microcontroller Group Users Manual*. 3. Jan. 2018. URL: https://www.renesas.com/us/en/doc/products/renesas-synergy/doc/r01um0001eu0140-synergy-s7g2.pdf?key=087dbe4a357a9c772717fce6ab75a16b.

[Ren18b] Renesas Electronics Corporation, Hrsg. *Renesas Electronics Corporation*. 2010-2018. URL: https://www.renesas.com/en-eu/ (besucht am 12. 05. 2018).

[rnw14] rn.wissen.de, Hrsg. *Timer/Counter (Avr)*. 2014. URL: https://rn-wissen.de/wiki/index.php?title=Timer/Counter_(Avr) (besucht am 16. 05. 2018).

[SKG18] Konstantin Schleser, Tim Krämer und Denis Graf. *Buffer-Overflow*. 2018. URL: https://www2.informatik.uni-hamburg.de/fachschaft/wiki/images/f/f0/7kraemer-Projekt-ausarbeitung.pdf (besucht am 05. 09. 2018).

[Sch18] Patrick Schnabel. *Raspberry Pi: GPIO - General Purpose Input Output*. 2018. URL: https://www.elektronik-kompendium.de/sites/raspberry-pi/2002191.htm (besucht am 21. 05. 2018).

[SGD09] Joachim Schröder, Tilo Gockel und Rüdiger Dillmann. *Embedded Linux. Das Praxisbuch*. 1. Auflage. Springer-Verlag Berlin Heidelberg, 15. Apr. 2009.

[SEG96] SEGGER Mikrocontroller GmbH, Hrsg. *J-Link/J-Trace User Guide*. 24. Mai 1996. URL: https://www.segger.com/downloads/jlink/UM08001_JLink.pdf.

[SEG15] SEGGER Mikrocontroller GmbH, Hrsg. *SWD*. 30. Sep. 2015. URL: https://wiki.segger.com/SWD (besucht am 25. 07. 2018).

[SEG18a] SEGGER Mikrocontroller GmbH, Hrsg. *J-Link-OB - Models*. 2018. URL: https://www.segger.com/products/debug-probes/j-link/models/j-link-ob/ (besucht am 19. 05. 2018).

[SEG18b] SEGGER Mikrocontroller GmbH, Hrsg. *J-Link/J-Trace Downloads*. 2018. URL: https://www.segger.com/downloads/jlink (besucht am 02. 07. 2018).

[SG17] Prof. Dr. Christian Siemers und Sebastian Gerstl. *Was ist ein Embedded System?* 27. Nov. 2017. URL: https://www.embedded-software-engineering.de/was-ist-ein-embedded-system-a-665424/ (besucht am 12. 05. 2018).

[Som12] Ulli Sommer. *Arduino. Mikrocontroller-Programmierung mit Arduino/Freeduino.* Franzis Verlag, 2012.

[Sta00] Richard Stallman. *The GNU Project.* 4. Dez. 2000. URL: https://noemalab.eu/org/sections/ideas/ideas_articles/pdf/stallman_eng.pdf (besucht am 03. 09. 2018).

[Sta+02] Richard Stallman, Roland Pesch, Stan Shebs und et al. *Debugging with GDB. Using the gdbserver program.* 14. Feb. 2002. URL: http://ftp.gnu.org/old-gnu/Manuals/gdb-5.1.1/html_node/gdb_130.html (besucht am 03. 09. 2018).

[Sti12] Michael Stilkerich. "Memory Protection at Option. Application-Tailored Memory Safety in Safety-Critical Embedded Systems". Dissertation. Universität Erlangen-Nürnberg, 2012.

[Str17] Iris Stroh. "Renesas Synergy-Plattform: Erhöhte Sicherheit und verbesserte Anbindung". In: *Markt&Technik* (11. Okt. 2017).

[Wal09] Roland Walter. *AVR-Mikrocontroller-Lehrbuch. Einführung in die Welt der AVR-RISC-Mikrocontroller am Beispiel des ATmega8.* Bd. 3. Auflage. Denkholz Buchmanufaktur, 1. Juli 2009.

[Waw09] Christian Walter Alois Wawersich. "KESO: Konstruktiver Speicherschutz für Eingebettete Systeme". Dissertation. Universität Erlangen-Nürnberg, 2009.

[WB05] Heinz Wörn und Uwe Brinkschulte. *Echtzeitsysteme. Grundlagen, Funktionsweisen, Anwendungen.* 1. Auflage. Springer-Verlag Berlin Heidelberg, 4. Apr. 2005.

[XJT18] XJTAG Limited, Hrsg. *Technischer Leitfaden für JTAG.* 2018. URL: https://www.xjtag.com/de/about-jtag/jtag-a-technical-overview/ (besucht am 04. 07. 2018).

Selbstständigkeitserklärung

Hiermit erkläre ich, dass ich meine Arbeit selbstständig verfasst, keine anderen als die angegebenen Quellen und Hilfsmittel benutzt und die Arbeit noch nicht anderweitig für Prüfungszwecke vorgelegt habe.

Stellen, die wörtlich oder sinngemäß aus Quellen entnommen wurden, sind als solche kenntlich gemacht.

Cora Mayerhofer

Mittweida, 14. September 2018